

## **General Disclaimer**

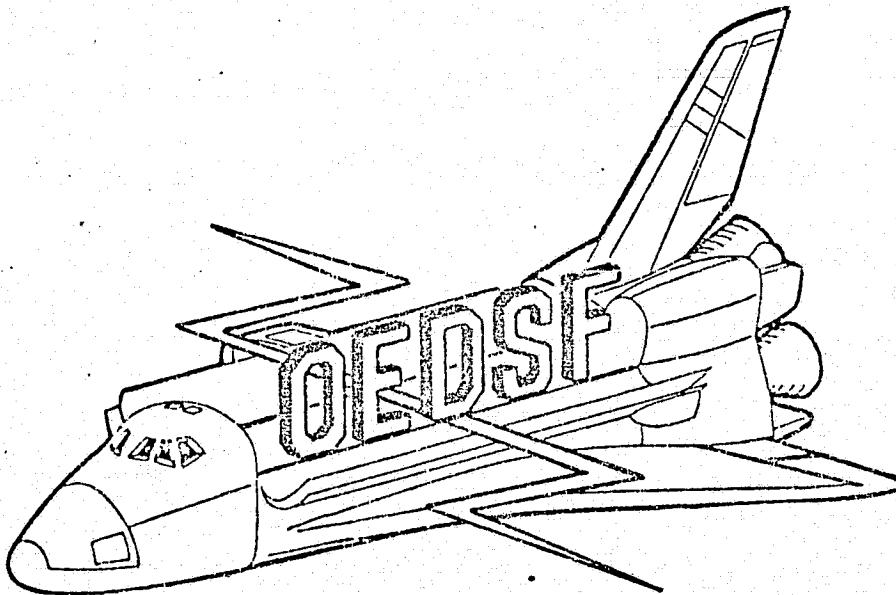
### **One or more of the Following Statements may affect this Document**

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.
- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.
- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.
- This document is paginated as submitted by the original source.
- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

JANUARY 1976

CR-151108

# ONBOARD EXPERIMENT DATA SUPPORT FACILITY --- TASK 2 REPORT



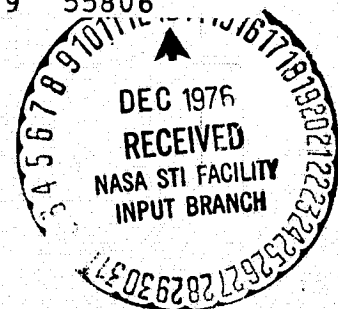
(NASA-CR-151108) ONBOARD EXPERIMENT DATA  
SUPPORT FACILITY. TASK 2 REPORT:  
DEFINITION OF ONBOARD PROCESSING  
REQUIREMENTS (General Electric Co.)  
HC A08/MF A01

N77-12120

166 p  
CSCL 09B G3/19

Unclas  
55806

Contract NAS 9-14651



GENERAL  ELECTRIC

JANUARY 1976

CONTRACT NAS9-14651  
DRL NO. T - 1122 (MA-461TA)

ONBOARD  
EXPERIMENT DATA  
SUPPORT FACILITY

TASK 2 REPORT

DEFINITION OF ONBOARD PROCESSING REQUIREMENTS

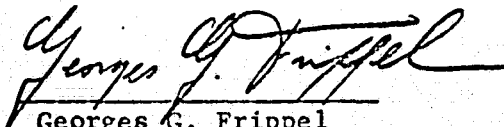
PREPARED FOR


NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

JOHNSON SPACE CENTER

HOUSTON, TEXAS

APPROVALS:

  
Georges G. Frippel  
Program Manager

  
Robert L. Giesecke  
Technical Monitor  
NASA-JSC Engineering and  
Development

PREPARED BY

GENERAL  ELECTRIC

VALLEY FORGE SPACE CENTER

P.O. BOX 8555

PHILADELPHIA, PA. 19101

## TABLE OF CONTENTS

	PAGE
1.0 INTRODUCTION	1-1
2.0 SUMMARY AND CONCLUSIONS	2-1
3.0 FUNCTIONAL FLOW DIAGRAMS	3-1
3.1 ATS	3-2
3.2 IRS	3-16
3.3 RADSCAT	3-30
3.4 CIMATS	3-34
4.0 OEDSF DEFINITION	4-1
4.1 ONBOARD/GROUND PARTITION	4-2
4.2 OEDSF REQUIREMENTS	4-10
4.3 GROUND REQUIREMENTS	4-14
4.4 TOTAL SYSTEM EVALUATION	4-20
5.0 OEDSF ARCHITECTURE	5-1
5.1 PROCESSING ARCHITECTURE	5-4
5.2 SYSTEM ARCHITECTURE	5-24
APPENDIX A - DECCOMPOSED ALGORITHMS	A-1
"    B - SOFTWARE/FIRMWARE/HARDWARE	B-1



## 1.0 INTRODUCTION

The Space Shuttle will accommodate 10,000 cubic feet of experiments and will fly on the average of 25 times per year. Typical payloads will collect on the order of  $10^{11}$  bits per day. The success of the STS missions requires that this data be handled and processed on a "routine" basis.

The OEDSF is a key step to the accomplishment of this requirement. The On-board Experiment Data Support Facility (OEDSF) will provide data processing support to various experiment payloads on board the Shuttle. The OEDSF study will define the conceptual design and generate specifications for an OEDSF which will meet the following objectives:

1. Provide a cost-effective approach to end-to-end processing requirements. The facility must provide a solution to the ever increasing costs of present ground system processing facilities within the context that flight hardware is inherently more expensive than ground hardware. It must be derived from a systems analysis of the end-to-end processing requirements and exploit the unique opportunities afforded by onboard processing and by the application of new technologies. These opportunities include adaptive control of sensors which collect the data; preprocessing of data using real-time available information such as ephemerides, spacecraft attitude and look angle, and atmospheric conditions as defined by auxiliary sensors; and the rejection of bad or useless data.
2. Service Multiple Disciplines. The design of the facility must consider the data gathering devices and the data processing requirements of the several disciplines which will utilize the STS. Since most shuttle flights will be interdisciplinary, the concept must be able to accommodate various mixes of these instruments and disciplines on the Space Shuttle.
3. Satisfy User Needs. The data should be immediately useful to the investigator. This implies a wide range of requirements corresponding to the spectrum of the user community. These range from those users who desire totally extracted information, to basic experimenters who need all pertinent data collected. The common thread linking all users is the set of criteria by which we evaluate all data: quality, timeliness, and cost.

4. Reduce the amount and improve the quality of data collected, stored and processed. The facility must help prevent bad or useless data from being collected and stored and reduce the amount of extraneous data which normally accompanies the useful portion of the collected data. It must provide for annotation and other useful formatting of the data.
5. Embody growth capacity. The facility will be capable of accommodating additional sensor groups derived from other disciplines, advances in the state of the art (second generation sensors) and expanding mission requirements. The facility will also be able to readily expand its own capabilities by providing for the accommodation of advances in technology pertinent to the facility's functions. This objective indicates a modular approach to the design of the facility.

This study is divided into four major tasks which are further divided into subtasks as described below. This report describes the effort performed in Task 2 and a part of Task 4.

#### TASK 1: Definition and Modeling of Classical Data Processing Requirements

Task 1 has defined the processing requirements for logical groups of experiments and was divided into two subtasks.

Subtask 1.1 - identified, tabulated, and characterized experiments which are candidates for STS missions. Based on these characterizations, "boundary" experiments were selected and their end-to-end data processing requirements defined. "Boundary" experiments are defined as those which impose demands on the system of such magnitude that their resolution will also satisfy the demands of many experiments whose requirements fall within the envelope defined by the boundary experiment.

Subtask 1.2 - was the combining of the boundary experiments into logical groups to permit viewing the OEDSF as an integrated system, and the definition of the groups' processing requirements. The results of this subtask were the end-to-end processing requirements for groups of experiments.

The output of Task 1 specified the end-to-end processing requirements for groups of experiments which represent boundaries on such requirements.

## TASK 2: Definition of Onboard Processing Requirements

Task 2 defines the onboard processing requirements for the grouped sensors and is partitioned into three subtasks.

Subtask 2.1 - accepts as input the definition of mission-oriented groups of sensors and their end-to-end processing requirements. An end-to-end functional flow diagram is generated using functional blocks to show the processing steps necessary to convert raw sensor data into usable information. Each block is studied to determine its potential for application of new techniques and processing alternatives; the results are stated in terms of algorithms and procedures. To a first-level approximation, the computation and storage requirements are also estimated to provide the basic tradeoff materials.

Subtask 2.2 - is essentially the rational choice of the space/ground partition line on the functional flow diagram produced in Subtask 2.1. It is an iterative decision based on system-level tradeoffs, modeling of the costs and performance of implementing each functional block on the ground or in space and the iterative feedback from Task 3 and Subtask 2.3. The major output of Subtask 2.2 is the definition of the processing requirements for the onboard portion of the total processing system. The Onboard Experiment Data Support Facility is defined in terms of architecture, processing requirements in terms of data and throughput rates, and identification of compatible processing techniques and equipment.

Subtask 2.3 - evaluates the effectiveness of a processing system. The effectiveness evaluation is used primarily to enhance the feedback from Task 3 in order to make better space/ground partition decisions in Subtask 2.2.

Another benefit of Subtask 2.3 is the early identification of critical factors which might become flexibility or growth-limiting.

The output of Task 2 is the set of requirements for the Onboard Experiment Data Support Facility and the selection of its architecture.

### TASK 3: Conceptual Design and Specification for an Onboard Processor

Task 3 produces the conceptual design and specifications for the Onboard Experiment Support Facility, and the evaluation of the facility in terms of the objectives. It is divided into three subtasks.

Subtask 3.1 - refines the system architecture derived in Task 2 and defines the detailed OEDSF architecture. To whatever extent necessary the results of Subtask 3.1 is fed back to Task 2 to modify and enhance the end-to-end system synthesis.

Subtask 3.2 uses the outputs of Subtask 3.1 to transform the architectural concept into a well defined processor design and specification. An initial point design is generated. The refinement of this point design and the definition of a second continue throughout the subtask. Critical components are identified, design tradeoffs are performed and resolved. The results of Subtask 3.2 is fed back to Subtask 3.1 and Task 2 for iteration toward an optimal overall design.

Subtask 3.3 - performs in-depth analyses of the processor point designs produced over the duration of the Task 3 effort. These analyses begin after a processor has been sufficiently designed and specified in Subtask 3.2. They either serve as documentation that a given processor point design meets all the objectives (including those of growth capability, flexibility

and cost-effectiveness) or point out in a constructive manner where the design should be enhanced.

The output of Task 3 is the conceptual design and specification for the OEDSF.

#### TASK 4: Cost Estimates and Development Plan

Task 4 produces cost estimates for the OEDSF and for ground systems with and without the OEDSF so that a quantitative evaluation of the cost benefits derived from the OEDSF may be derived. It also produces a development plan showing the schedule phases required to develop the OEDSF. Task 4 is divided into two subtasks.

Subtask 4.1 generates the information specifically needed for cost estimating purposes which would otherwise be unnecessary to the design of the OEDSF. This includes increased details in certain areas of the functional flow diagrams, and definition of the ground segment of the system.

Subtask 4.2 generates a preliminary OEDSF development plan and determines the cost of the OEDSF its end-to-end data system, and a typical data system not based on onboard processing.

The output of Task 4 will be detailed cost estimates for the design, fabrication, qualification, and test of the OEDSF for its end-to-end system, and for an end-to-end data system without an OEDSF.

## 2.0 SUMMARY AND CONCLUSIONS

The major effort of Task 2 was the development of the functional flow diagrams for the six boundary experiments selected in Task 1. These experiments were:

- Advanced Technology Scanner (ATS)
- Infrared Spectrometer (IRS)
- Radiometer/Scatterometer (RADSCAT)
- Correlation Interferometry for the Measurement of Atmospheric Trace Species (CIMATS)
- Electron Accelerator
- Optical Band Image and Photometer System (OBIPS)

Other key tasks performed were the partitioning between onboard and ground processing, the evaluation of the benefits provided by onboard processing, and the selection of the OEDSF architectures.

The development of the flow diagrams for the Electron Accelerator and the OBIPS was halted as a result of difficulties discussed below.

The Electron Accelerator was primarily selected because it offers the opportunity for interactive operation with crew members. This interaction occurs as a result of the display of operating characteristics (such as I/V curves) which guide the operator's selection of operating modes. There is no technical difficulty in implementing these processes which are more in the nature of signal conditioning than signal processing. The problem lies in that the specific items to be displayed as well as the presentation desired have not been defined. We consider the processes required to be simple and will incorporate this capability into the OEDSF when the requirements are defined. The deletion of this capability at this time does not impact the design of the remainder of the OEDSF; its subsequent addition should have minimal impact on the OEDSF.

The OBIPS was selected because it is representative of experiments which collect a great deal of informationless data which, if edited out, can reduce the data recorded or transmitted by as much as 95%. A survey of available techniques for data editing disclosed that there presently exists no technique capable of identifying the presence or lack of information in video images except in very specific and rigidly controlled conditions. The development of such techniques is an effort worthy of a separate study and well beyond the scope of the OEDSF study.

The functional flow diagrams based on a real time processing implementation were completed for the remaining boundary sensors. Each function was evaluated with respect to its allocation on board or on the ground. During this activity certain processes were modified to permit their performance in instances where this location on board was deemed beneficial from a system standpoint. Two examples are given:

- 1) The ATS GPC correlation was changed to a predictive process using a Kalman Filter. The alternate approach is to store an entire scene in seven spectral bands, requiring close to  $10^9$  bits of memory.
- 2) The IRS present process substitutes yesterday's temperature at a point when today's measured value cannot be obtained due to clouds. This process requires a daily updated storage of yesterday's temperature for every point. The proposed approach is to use the average of the four nearest valid neighbors around the point.

Simultaneously with this effort, a set of criteria for partitioning the system into an onboard segment and a ground segment was developed. The results of these activities were the partitioning decisions discussed in Section 4.1.

The set of requirements for the OEDSF were developed from the onboard segment of the flow diagrams. These requirements form the basis for the architecture tradeoffs described in Section 5, and for the conceptual design of the OEDSF which will be performed in Task 3.

The processes defined by the onboard segment of the boundary sensors were decomposed into algorithms which define the OEDSF functions. These functions may be recombined to produce a far greater number of processes than those required by the boundary sensors.

The four sensors produced a basic set of 18 functions in four categories:

- . Trigonometric
- . Exponential (and logarithmic)
- . Algebraic
- . Control

The ground segment was examined to insure that the partitioning decisions and the selected onboard processing techniques did not increase the ground requirements so as to negate the gains derived from onboard processing.

Only the IRS processing created an additional task on the ground. This task is essentially negligible.

The evaluation of the resulting system was performed on a qualitative basis. A quantitative evaluation of the cost benefits will be performed in Task 4 when the costs associated with the OEDSF are determined.

The evaluation indicates significant advantages derived by the OEDSF. The cost effectiveness will be determined by the costs of the OEDSF. It also uncovered areas related to the integration and testing of instruments with OEDSF. This potential problem area must be addressed and resolved if the OEDSF is to realize all its potential benefits.

Architectures were selected for the system and the processing levels based on the OEDSF requirements and the preliminary implementation tradeoffs.



The system architecture is a structural architecture; that for the processing (subsystem) level is an array.

There are many far-ranging benefits to be derived from onboard processing, but they will be realized only through judicious designs and proper planning.

The cost of the OEDSF must be balanced by significant cost reductions in ground systems. These include the cost of equipment, programming and operation.

The OEDSF creates additional benefits by virtue of its requirements: better discipline, better planning, specialized programming, and its status as Flight Equipment.

The balance of this report is organized as follows:

Section 3 contains the functional flow diagrams of the boundary sensors with a summary of the sensors' operation.

Section 4 contains the definition of the OEDSF and describes the Onboard/ Ground partitions and their rationale, the OEDSF requirements and the ground requirements resulting from the partitions, and an overall system evaluation based on these requirements.

Section 5 is a discussion and trade-off of applicable architectures for the OEDSF which satisfy the requirements derived in section 4.

Appendix A contains the algorithms resulting from the decomposition of the functions derived in section 3.

Appendix B is a generalized discussion of the characteristics of software, firmware, and hardware.

### 3.0 FUNCTIONAL FLOW DIAGRAMS

This section contains the functional flow diagrams for the four boundary sensors. Each flow diagram is preceded by a short summary of the sensor's operation from a processing viewpoint, and a discussion of the flow diagram with the rationale for the various trade-offs. (A more complete description of each sensor is contained in the appendices to the Onboard Experiment Data Support Facility Task 1 Report, dated September 1975).

Trade-offs between hardware, firmware, and software are based on specific factors discussed in the write-up, or on general factors which are discussed in Appendix B.

A detailed processing requirements diagram indicating the required functions in closed form, and the source of constraints and coefficients also precedes the IRS & CIMATS flow diagrams. These requirements are based on the methods presently used in the all-ground approach, and were changed in the flow diagrams as required to exploit the onboard features of the OEDSF. In particular, the processes were converted to real-time, and techniques suited to this approach substituted for batch-oriented techniques.

The flow diagrams reflect the time dependence of each of the processes and incorporate time delays and buffer storages as required.

### 3.1 ADVANCED TECHNOLOGY SCANNER (ATS)

This section describes the functional flow developed for the ATS, and the rationale for the various trade-offs effected.

#### Functional Flow Description

The functional flow diagram developed for the Advanced Technology Scanner, is discussed with reference to Figure 3.1-4. The sensor output establishes the OEDSF processor and sensor interface. This flow has been developed for a real time system; therefore certain additional requirements must be incorporated into the system. The reader is referenced to appendix A of the Task 1 report for a detailed description of the ATS's operation.

The Advanced Technology Scanner is a high resolution multi-spectral scanner with a high frequency composite data rate and a medium to low frequency equivalent channel rate. It is assumed that the sensor provides a channel output format similar to the channel output format for the Landsat sensors. This format is shown in Figure 3.1-1.



Single Channel Sensor Output

Figure 3.1-1

Based on this format, the sensor provides an 80-20 duty cycle comprised of 3730 picture elements and 932 calibration words. The calibration wedge is generated on alternate mirror sweeps and applies to the preceeding and succeeding mirror sweeps. This calibration wedge is generated on the mirror re-trace by sampling the internal lamp. The position of the calibration wedge impacts the process and is discussed in the explanation of the functional flow diagram.

Independent of the process location, i.e. space or ground, some general characteristics may be developed from the physical operation. The number of pixels per mirror sweep is a function of the mirror sweep frequency, the mirror profile, the number of bits per pixel, and the sample rate of the detector output. This may be expressed as

$$P = \frac{T_m + \Delta T_m}{T_{PCM}} \frac{1}{N}$$

where

P = number of pixels

T<sub>m</sub> = the mirror sweep frequency

ΔT<sub>m</sub> = the mirror profile (The variances in the mirror sweep rate)

T<sub>PCM</sub> = the sample rate

and

N = the number of bits per pixel

The maximum number of samples per scan line is 3730 pixels; a scene is comprised of 3730 scan lines. Based on the sensor description, the data will be processed on a line per detector basis so that the machine data rates may be computed as

T<sub>p</sub> = 8.0 microseconds

T<sub>p</sub> = Time duration of 1 pixel

T<sub>l</sub> = 29.84 milliseconds

T<sub>l</sub> = Time duration of 1 line

and

T<sub>s</sub> = 1.855 minutes

T<sub>s</sub> = Time duration of 1 scene

If the processing is performed on a band parallel, channel sequential, pixel sequential basis, some form of intermediate storage is required at the input of the OEDSF processor. The size of the storage is determined by the processing techniques and the characteristics of the output mass storage device. If it is assumed that the output recorder has a capacity to store data at a rate of 2.0 megabits/sec per track and has 120 tracks, the storage rate is 240 megabits/sec but the data may be operated on in parallel so that operation time is reduced without the use of complex registers and multiplexers.

Based on the high density tape recorder characteristics using 16 tracks per band, the data rate of the processor may be 4 megabits per second.

A mirror sweep of data for a single spectral band is twenty scan lines, or a storage capacity of 596.8 kilobits. If a FIFO type buffer is used at the maximum permissible clock rate, the total storage capacity required per spectral band is 238.72 kilobits. By using this technique the sensor interface is simplified and the OEDSF processor machine time is within the current state-of-the-art.

The functional flow diagram is discussed in the light of real time processing based on a brief description of the state-of-the-art of machine time and the sensor interface. Initially, data from the sensor must be demultiplexed into a calibration data stream and a spectral or image data stream. Since the calibration data is located between the two scan lines to which it implies, a two mirror sweep delay must be incurred initially by the spectral data. In addition, a filter implemented in the correction process requires an additional two mirror sweep delay so that the total spectral data delay required is equal to 4 mirror sweeps. and is shown as process 8 .

The initial requirement for the processing of the calibration wedge is to isolate a predetermined threshold within the calibration wedge as shown in process 1 . When the threshold is detected a calibration sample is removed from the data stream in process 2. The sample consists of six bytes of calibration voltage and six bytes of a corresponding radiance measured from the position (word count) of the threshold detected. Based on the actual relationship of the location of the radiance and the voltage some small delay may be required for the next actual process. In particular, the magnitude of the delay is dependent on the speed at which

processes 3, 4 are performed.

Having selected the calibration voltage and radiances, the radiance is initially normalized to the maximum specified radiance for a given spectral band in process 3. In order to remove detector degradation, linear regression is used since the ideal radiance-voltage relationship is linear. The regression coefficients are computed in process 4. If an increased accuracy is required the sample may be increased from six points to nine points with the additional three sample points located around the center of the line. Figure 3.1-2a depicts the theoretical relationship; Figure 3.1-2b depicts the current techniques; Figure 3.1-2c is an improved technique with no significant system impact.

Based on the regression coefficients determined in process 4, the initial gain and offset for the correction coefficients are computed in process 5 in conjunction with the sampled calibration voltages. These values may contain noise, and therefore vary, so a Kalman Filter is used because the optimum variable weighting factor reduces signal variance due to noise very rapidly and allows reasonable initial transient response time. The baseline gains and offsets are processed in the Kalman Filter, process 6. Since the filter requires the previous gain and offset, a two mirror sweep delay, process 7, is required. An additional two mirror sweep delay is required for the spectral data. Consequently, the radiometric calibration coefficients are computed in process 9 so that an additional delay equivalent to the process times required for processes 1, 2, 3, 4, 5, 6, and 9 must be incorporated into the system. The magnitude of the delay at a machine cycle time of 250 nanoseconds is of the order of 32 picture elements. Based on the machine pixel period of 250 nanoseconds, the radiometric coefficient calculations must be computed within 1.865 milliseconds. Further, assuming 50 nanoseconds per basic operation 37.3 kilo operations can be performed. From the functional flow diagram, the radiometric coefficient computation requires approximately 100 complex operations. The radio-

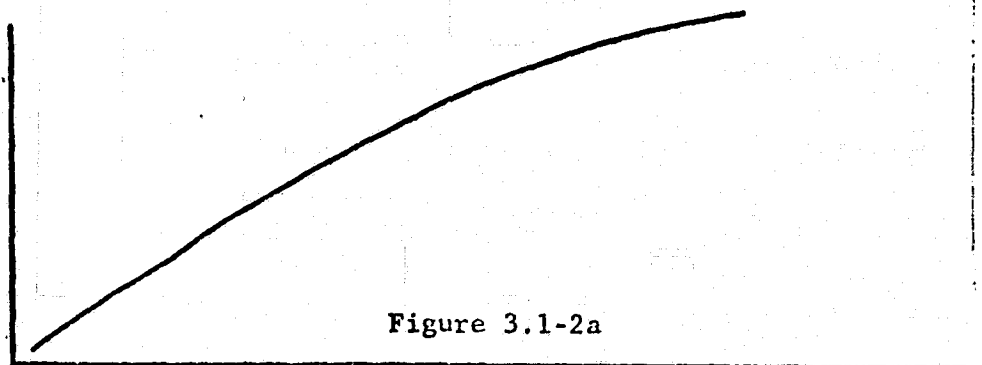


Figure 3.1-2a

Theoretical

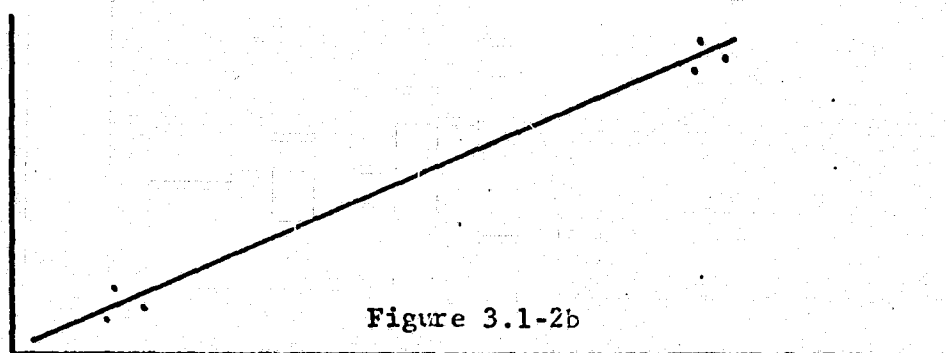


Figure 3.1-2b

Current

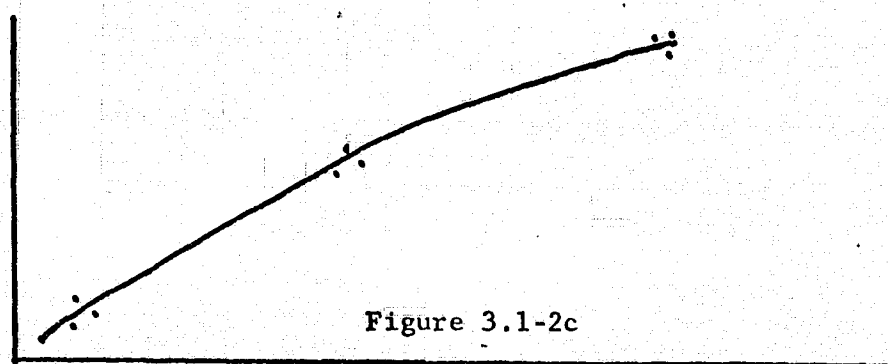


Figure 3.1-2c

Improved

metric correction processes shown in the functional flow diagram in processes 1 through 10 yield a radiometric correction that is accurate to within two quantum levels. The spectral data sampled is corrected in process 10 based on the linear relationship previously stated for a detector. An alternate approach which reduces the computation required is to apply a nominal correction to the entire scene. This approach yields a radiometric correction which does not satisfy the majority of the investigators.

The second portion of the Advanced Technology Scanner discussed is the geometric correction process. The initial geometric correction process discussed is the model processor, process 19, because its operation is reflected in the remainder of the functional flow diagram.

Depending on the particular sensor used, additional corrections may be required, such as the line geometry for the conical scanner. For satellite-sensed images and spacecraft images, two basic distortions are incurred. Those systematic distortions due to earth effects and the scanner and scene distortions due to vehicle dynamics. Therefore, certain parameters are required either a priori or on a dynamic basis at a scan line or scene rate.

Shuttle GNC data or an accurate gyro platform must provide yaw, pitch, and roll for the OEDSF processor. Since synchronization of these two processors is not likely, the OEDSF processor will sample the guidance platform output at the required time.

In addition to the yaw, pitch, and roll of the orbiter, the GNC data must include mean equatorial time, longitude, latitude, sun angle, attitude and velocity.

The Advanced Technology Scanner must provide line number, pixel number, platform look angle, and the spectral band number synchronously with the data. Finally ancillary data must provide the mirror sweep rate, the mirror profile, and sun disk values and previous thresholds.



Having established these parameters, process 19 computes the correction vectors depicted. The correction is computed in the following manner: Based on the parameters, the error vectors are computed and their inverse taken. For example, if a negative yaw angle is measured the correction vector is positive. In order to correct the image, certain parameters must be known with respect to the earth's surface; a coordinate system must be selected. There are three major coordinate systems employed today with forward and reverse transforms to change from system to system. The system used is the Universal Transverse Mercator (UTM) because it is the most widely utilized system.

Process 19 removes the systematic and scene distortions as shown in the functional block. From the longitude and latitude, the OEDSF processor determines a set of ground control points from the data base and translates these parameters to line, pixel, and scene numbers. The appropriate ground control points and their locations are transferred from a GCP library to the GCP reference (process 17) prior to active scanning. Based on the line and pixel numbers, the GCP controller, process 18, derives the location of the GCP search area in the image and selects the reference GCP from the data base 17.

Having established this parameter interaction, the search area is selected from the image by the strip, process 11, and an image data array formed by process 12. This sub-image is further processed by a two dimensional Hadamard Transformation to obtain its frequency spectrum. The Hadamard Transformation, process 14, was selected in favor of the conventional Fourier Transformation for the following reasons: 1) The Hadamard Transformation is easily implemented with either random logic or with a digital computer while the Fourier Transformation is oriented to a software implementation on a medium size digital computer and not with random logic. 2) A Hadamard Transformation which is  $N \times N$  results in an  $N^2$  array while, a Fourier Transformation which is  $N \times N$  results in a  $2N^2$  array. 3) The Hadamard Transform, being a rectangular expansion, is binary in nature so that the trans-

formation matrix is  $2(N \times N)M$  bits where  $M$  is the number of bits in the trigonometric functions. For an eight bit picture element  $M$  is eight.

Having obtained the Hadamard Transform, the frequency spectrum of the search area is correlated with the reference and the GCP located. If the GCP is located in the correlator, process 15, the model processor is notified and the next area selected. The process flow shown in Figure 4 requires the image to be delayed since the distortion being computed is for the scene from which the GCP's are extracted. Consequently, a scene delay must be accomplished by process 13 and for optimal machine times is still far beyond the state-of-art for any type of delay line either acoustical or electrical.

An alternate process has been implemented. This technique eliminates the requirement for the delay line while maintaining accuracy to acceptable levels. The alternate process is based on prediction or estimation theory and requires a Kalman Filter. The order of the filter is dependent on the accuracy requirement and with a minimum filter yields an overall error of 0.5 pixel rms. This technique is implemented in the functional flow in the following manner: A Kalman Filter is incorporated into process 19. The active scanning is initiated at least two scene times early (the first two scenes do not receive a precision correction). Based on the distortion determined in these scenes, the distortion present in the next scene is estimated, and a correction factor is applied. The accuracy is determined by the number of ground control points used and the number of scenes. Consequently, the signal flow diagram for the baseline functional flow diagram shown in Figure 3.1-4e reduces to the signal flow diagram shown in Figure 3.1-4. The significance of this approach will be explained in the later paragraphs.

At the present time, the number of ground control points required ranges from three per scene to ten per swath. It is assumed that only three ground control points per scene will be required by the 1980 time frame due to increased sophistication of processing techniques and instrumentation.

The final step in the functional flow is the geometric correction and resampling process. Since the correction or more precisely the distortions are orthogonal, the correction may be applied sequentially to the image. Initially, the data is corrected along the scan line (x direction) to obtain a set of uniformly space pixels. The correction process, process 20, is an interpolation i.e. filter, based on selected picture elements and a set of filter weights. Since the scan line is operated on asynchronously, some temporary storage is required and has been determined to be a maximum of eight scan lines.

The model previously described computes the location of the input domain pixels in the stored scan line and their respective weighting functions. Process 20 performs the actual resampling. Having corrected the image for along the line errors, the image is corrected for across the line errors in an identical manner except that process 19 computes the correction vector in the y direction given the x direction. The result is a set of uniformly spaced picture elements registered to the earth's surface in a defined coordinate system.

The size of temporary storage for the across the line correction is a function of the yaw angle of the vehicle and the number of picture elements in the output domain scan line. Assuming a yaw angle of 0.5 degrees, 3730 pixels in the output domain scan line, and a four point sample, a temporary storage of 32 scan lines is required.

If a precision correction is to be applied. A four point sample is used for the correction based on a  $\sin x/x$  filter. If no precision correction is to be applied, The value of the nearest neighbor is used so that original data may be reconstructed by the investigator for special information extraction and analysis. By using a prediction technique additional flexibility is achieved in that the image may be processed with no correction. Systematic correction, or scene correction may be obtained from the same model without any modification.

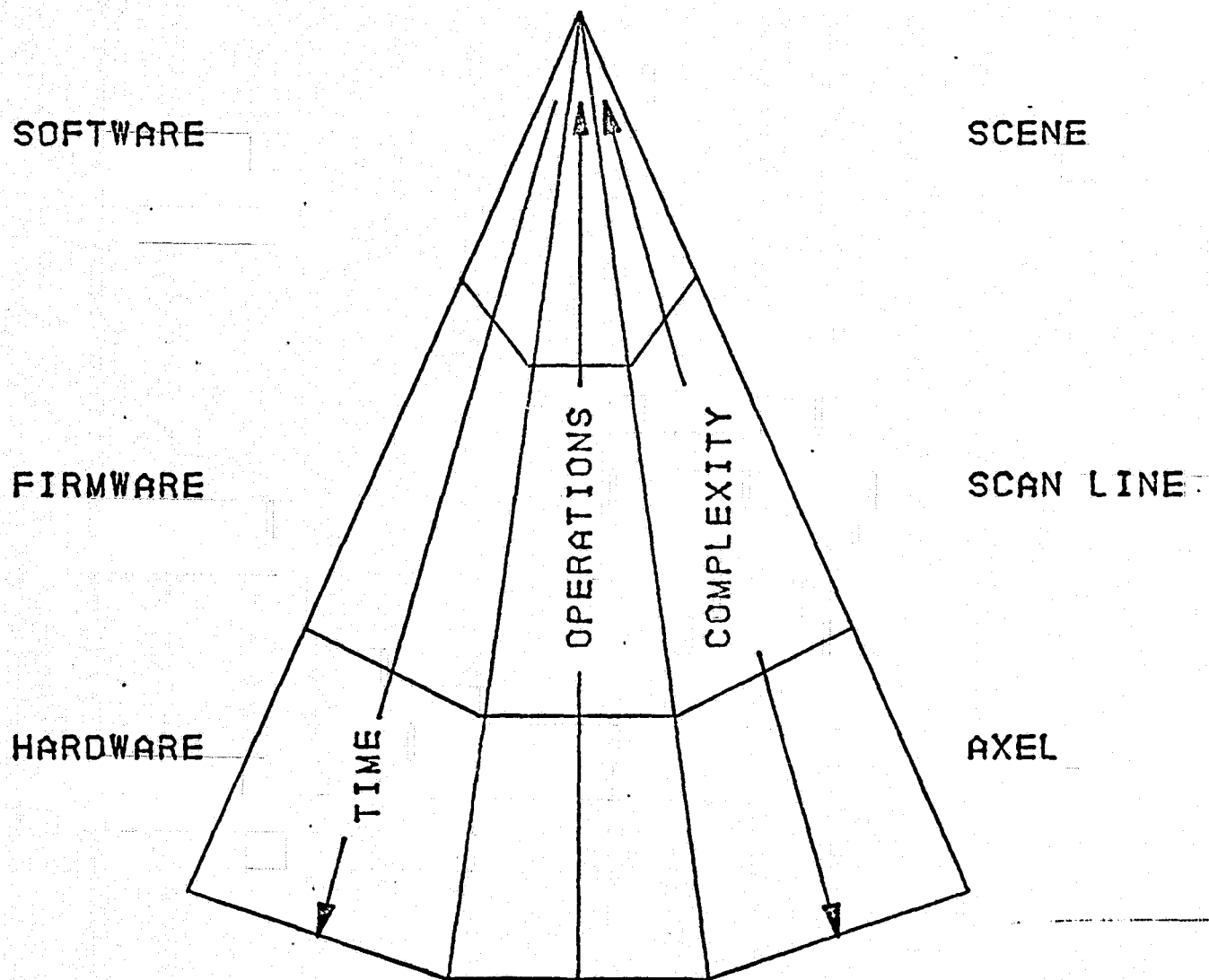
The final process, process 22, is the capture of the image on some mass storage device for user dissemination or transmission. The discussion has been with respect to a single spectral band. The total or composite signal flow diagram is shown in Figures 9 and 10. Since the process performed in step 19 applies to the image and the band to band relationship is a function of the sensor geometry and operation, process 19 is only required for a single band and the band to band transformation is included in the process.

#### Hardware/Software/Firmware Trade-off

The required processes must be partitioned into either hardware, software, or firmware based on the functional flow diagram. In order to achieve this partitioning, criteria must be established and applied to the general processes so that the preliminary architecture may be developed. General guidelines are discussed in Appendix B. For the Advanced Technology Scanner, the basic criteria employed for this partitioning are the periods of time within which a function must be performed, the complexity of the function, and the level of application i.e. pixel, scan line, or scene as well as the frequency of application. Initially, a basic unit must be established which will be a scene (an array of 3730 by 3730 picture elements). Pictorially, the partitioning criteria form a pyramid shown in Figure 3.1-3

# HARDWARE/SOFTWARE/FIRMWARE PYRAMID

FIGURE 3.1-3



Each process has been labeled as to its tentative method of implementation.

Each process that must be performed at a pixel rate has been basically assigned a hardware implementation; each process that must be performed at a scan line rate assigned a firmware implementation, and scene rate processes assigned a software implementation. Based on a machine pixel period of 250 nanoseconds a scan line period is 1.1655 milliseconds and a scene duration is 5.4335 seconds.

Since the real time pixel rate is 8.0 microseconds, a scan line period 29.84 milliseconds and a scene period 1.85 minutes, significant processing time is available for other functions or other sensor processing within the OEDSF processor.

Only those functions which are candidates for trade off are discussed.

Since the threshold detector and the sampler must operate on the data at a pixel rate, these devices are assigned a hardware implementation. Having obtained the respective samples of radiance and voltage, processes 3, 4, 5, 6, 7, and 9 can be implemented in either hardware or firmware because of the scan line duration. Process 3 is implemented in firmware because the values change on an infrequent basis but are alterable in nature. In addition this process must be performed at the pixel rate; therefore software is not capable of handling the time requirement. To perform this function in software would require a complex subroutine that at best in 1980 would require approximately 57.7 microseconds which equates to an order of 5 reduction in current microprocessor computation times. Since a scan line duration is 1.1655 milliseconds and the division must be performed nine times, process 3 requires approximately 44.4 percent of the available time. Performing this function in firmware, eg look up table, at the pixel rate requires 0.19 percent of the available time. Since only an eight bit picture element is used, a single integrated circuit look up table of 256 x 8 would be required.

Process 5 has been initially assigned a firmware approach but may be also implemented in hardware depending on the development of a Programmable Logic Array (PLA).

If implemented in hardware this process would require the execution of forty-two instructions or a delay of 9 pixel times. If implemented in firmware, a processor would be required and, based on the reduction of 5, would require approximately 1.33 milliseconds which exceeds the scan line duration. Consequently, this process is assigned to hardware.

Since the functions required for process 4 are identical to process 5, a hardware implementation is the selected candidate if the programmable logic array is available since only two integrated circuits are required. The process could be implemented in firmware but requires 88.8 percent of the allotted time.

Process 6 is a Kalman Filter which will be implemented in software since 117 microseconds are available and the hardware does not allow for variable weighting factors. The next process is basically software in nature. Based on the same assumptions this process requires 172.5 microseconds or 14.8 percent of the available time. Consequently, based on these assignments, the required coefficient processes require less than 50 percent of the available time.

The remaining process 10, must operate on the spectral data at a pixel rate, so that a hardware assignment is required. The remaining processes are assigned to hardware if they must be performed at the pixel rate, firmware if they are to be performed at the scan line rate and software if they are to be performed at a scene rate with the exception of the Hadamard Transform.

This process has been assigned to hardware because a minimum of three transforms per scene is required. The transform is not a difficult random logic implementation

and is simplified by the assumption that the intensities in the image data array will not vary more than 30% over 20 pixels direction. Based on this assumption and given a symmetrical array, the kernel is separable and the process is reduced to two sequential operations.



ORIGINAL PAGE IS  
OF POOR QUALITY

ADVANCED TECHNOLOGY SCANNER  
FINAL FUNCTIONAL FLOW DIAGRAM FOR  
REAL TIME

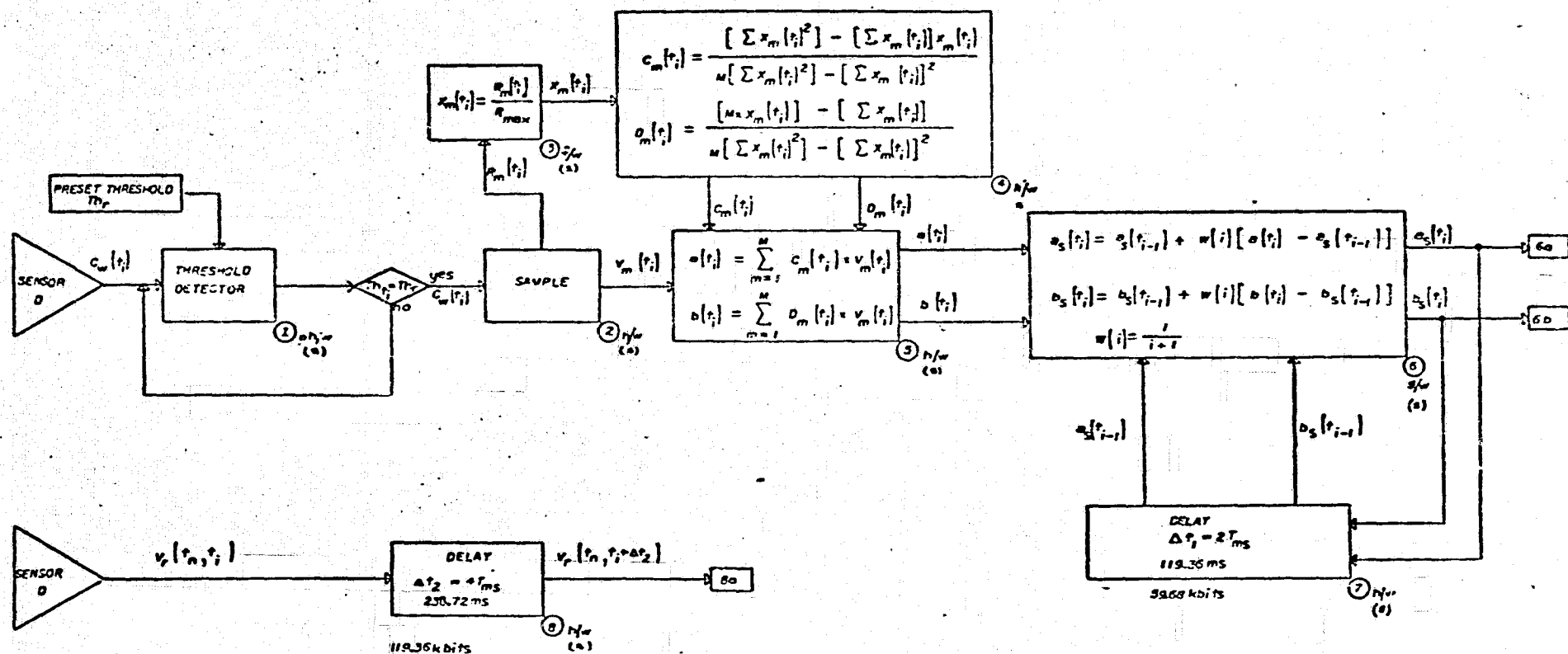


Figure 3.1-4a



ORIGINAL PAGE IS  
OF POOR QUALITY

ADVANCED TECHNOLOGY SCANNER  
FINAL FUNCTIONAL FLOW DIAGRAM CONTINUED  
FOR REAL TIME

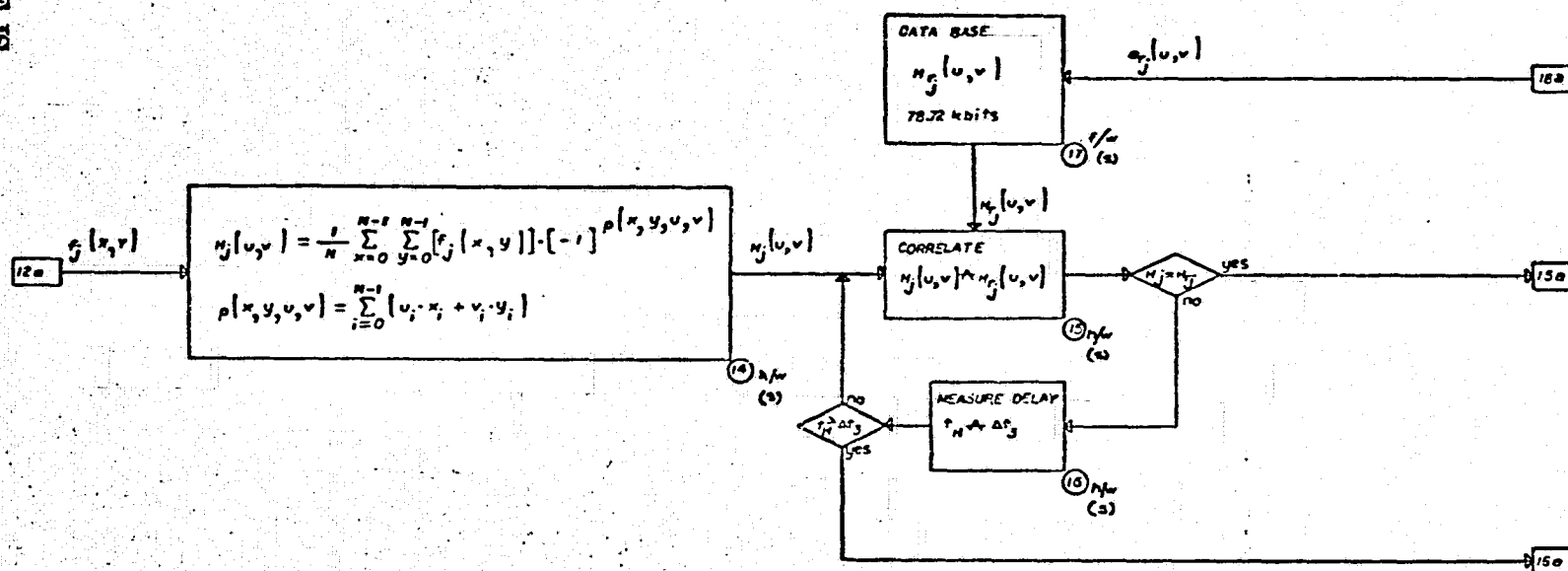
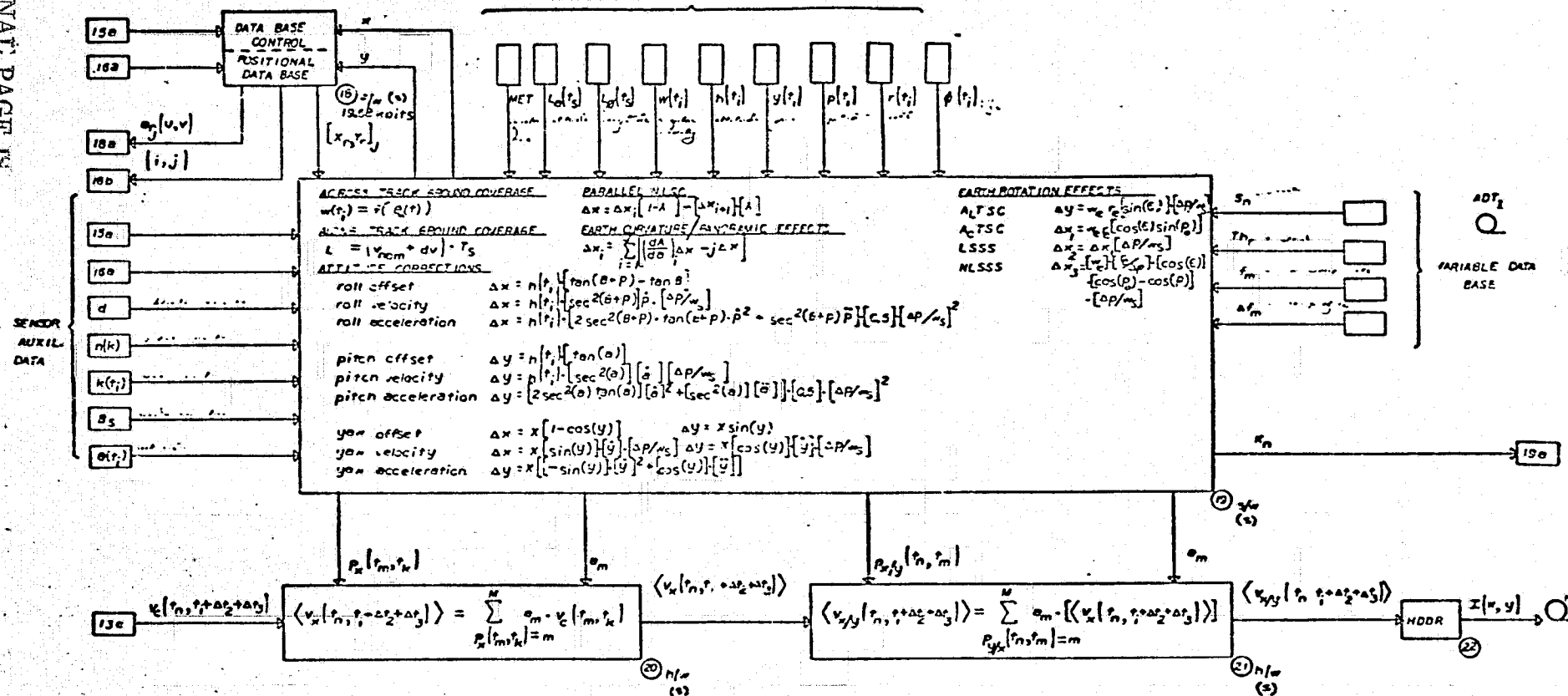


Figure 3.1-4c

# ADVANCED TECHNOLOGY SCANNER FINAL FUNCTIONAL FLOW PROGRAM FOR REAL TIME

SPACECRAFT 68C GENERATED DATA



ADVANCED TECHNOLOGY SCANNER  
STANDARD MODEL SIGNAL FLOW DIAGRAM GENERAL

ORIGINAL PAGE IS  
 OF POOR QUALITY

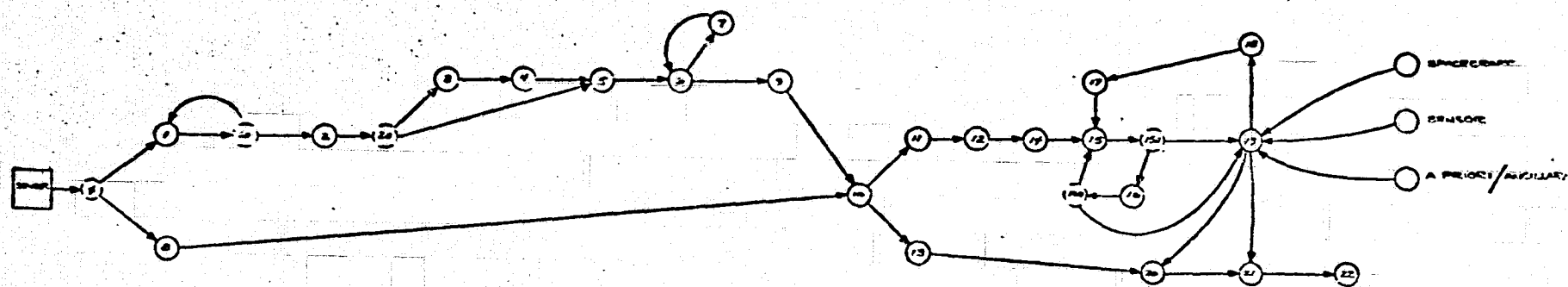
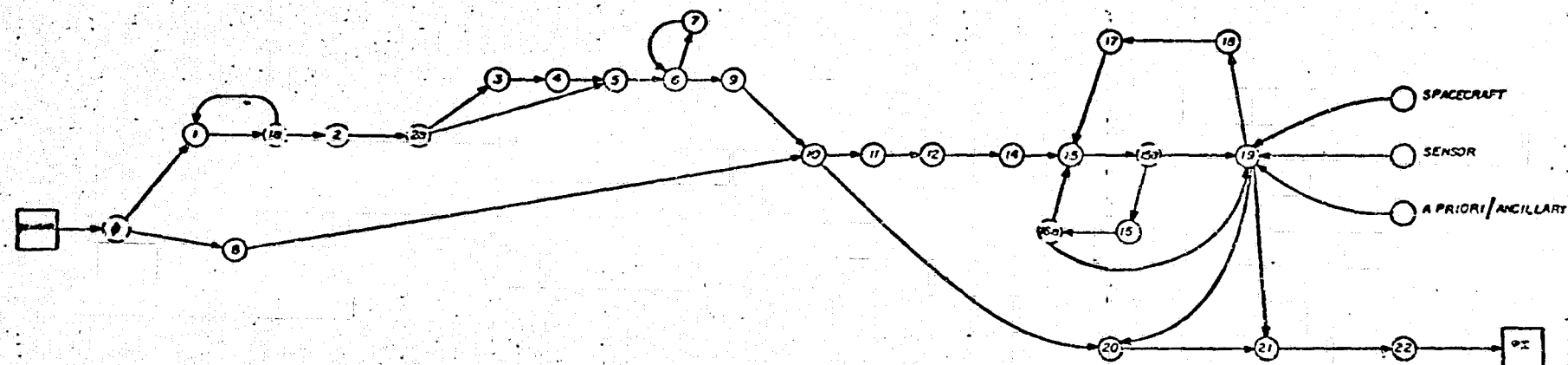


Figure 3.1-4a

ADVANCED TECHNOLOGY SCANNER  
PREDICTION MODEL SIGNAL FLOW DIAGRAM GENERAL

REPRODUCED FROM THE  
 OFFICIAL RECORDS  
 OF THE ARMY



ADVANCED TECHNOLOGY SCANNER  
STANDARD MODE: COMPOSITE SIGNAL FLOW DIAGRAM

ORIGINAL PAGE IS  
 OF POOR QUALITY

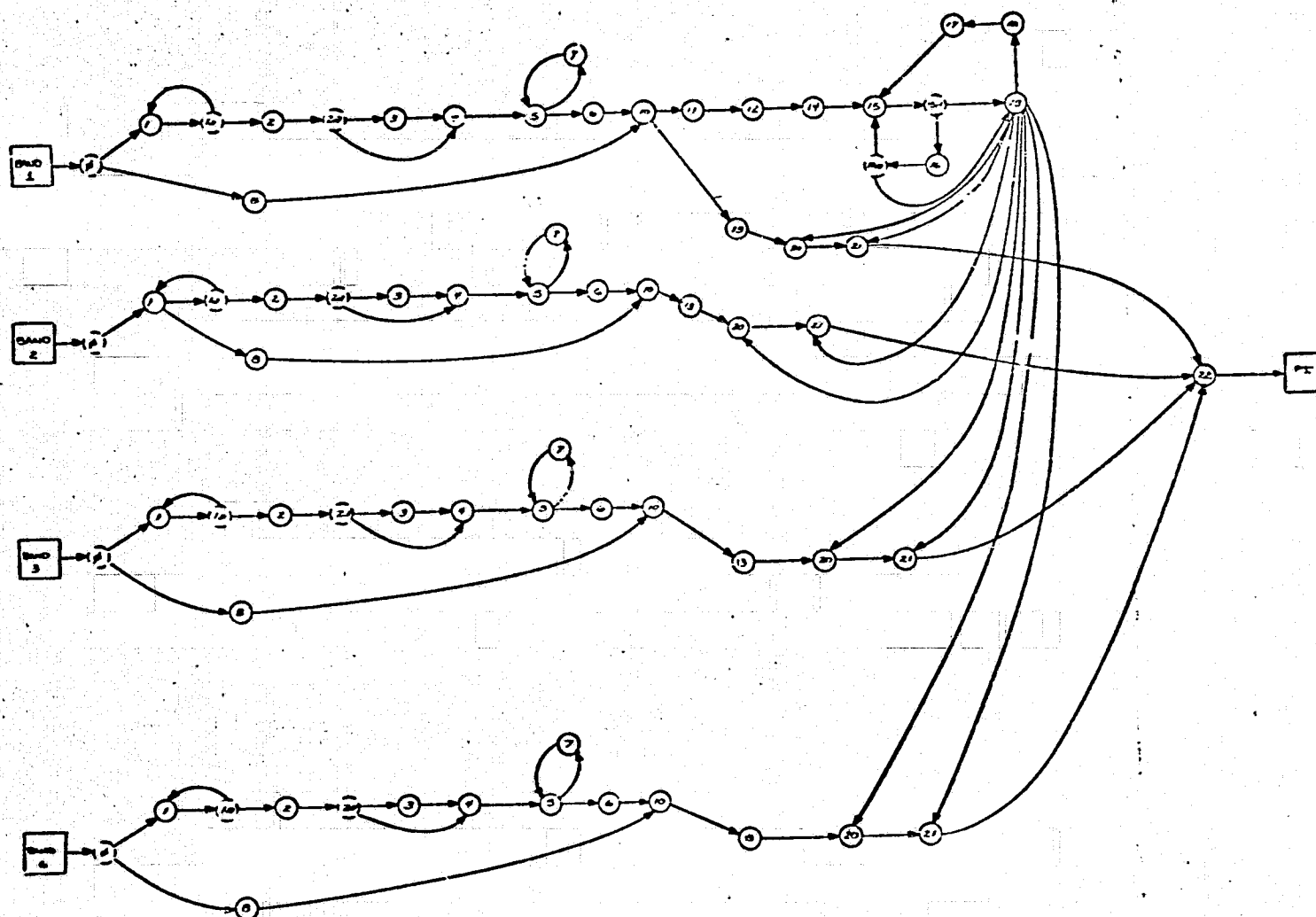


Figure 3.1

### 3.2 INFRARED SPECTROMETER (IRS)

This section describes the functional flow developed for the IRS and the rationale for the various tradeoffs effected.

#### Introduction

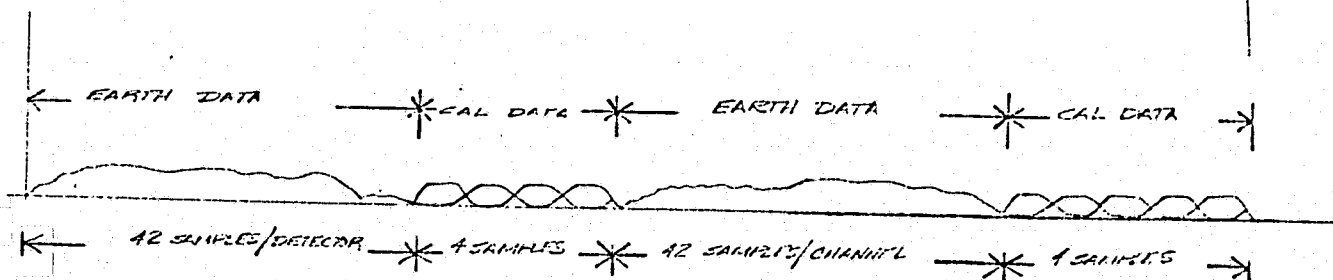
The Infrared Spectrometer is an instrument developed to determine the thermal structure of the earth's atmosphere. The sensor is a multi-detector based on a filter wheel which scans about the nadir normal to the orbit plane. A more complete description of the IRS is contained in Appendix B of the OEDSF Task 1 Report. As the sensor mirror is rotated via a stepping motor through the nadir, a sample of the dwell of each step is digitized and transmitted as the sample of detected infrared energy synchronously with mirror position. After completing the prescribed transversing, electronic calibration data is measured to complete one mirror sweep (or scan line). This process is repeated for twenty mirror sweeps at which time two internal black bodies and space are sampled to establish temperature reference points. One map is comprised of twenty-three scan lines and is termed a grid. The resulting sensor output waveforms are shown in Figures 3.2-1 and 3.2-2.

The channel frequency for the IRS as determined by the mirror rate is 3.39 kilobits per second with the earth information constituting 84.7% of the scan line.

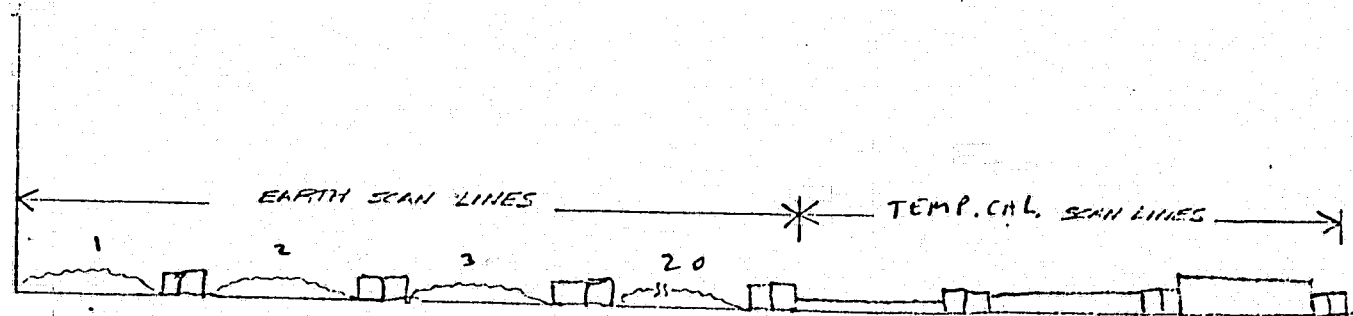
#### Information Processing

Figure 3.2-8 is the set of processes required for the IRS data. A top-level version is shown in Figure 3.2-3. Figure 3.2-9 is the functional flow diagram developed for the IRS. The processing of the IRS data may be divided into functional categories





SCAN LINE FORMAT  
FIGURE 3.2-1

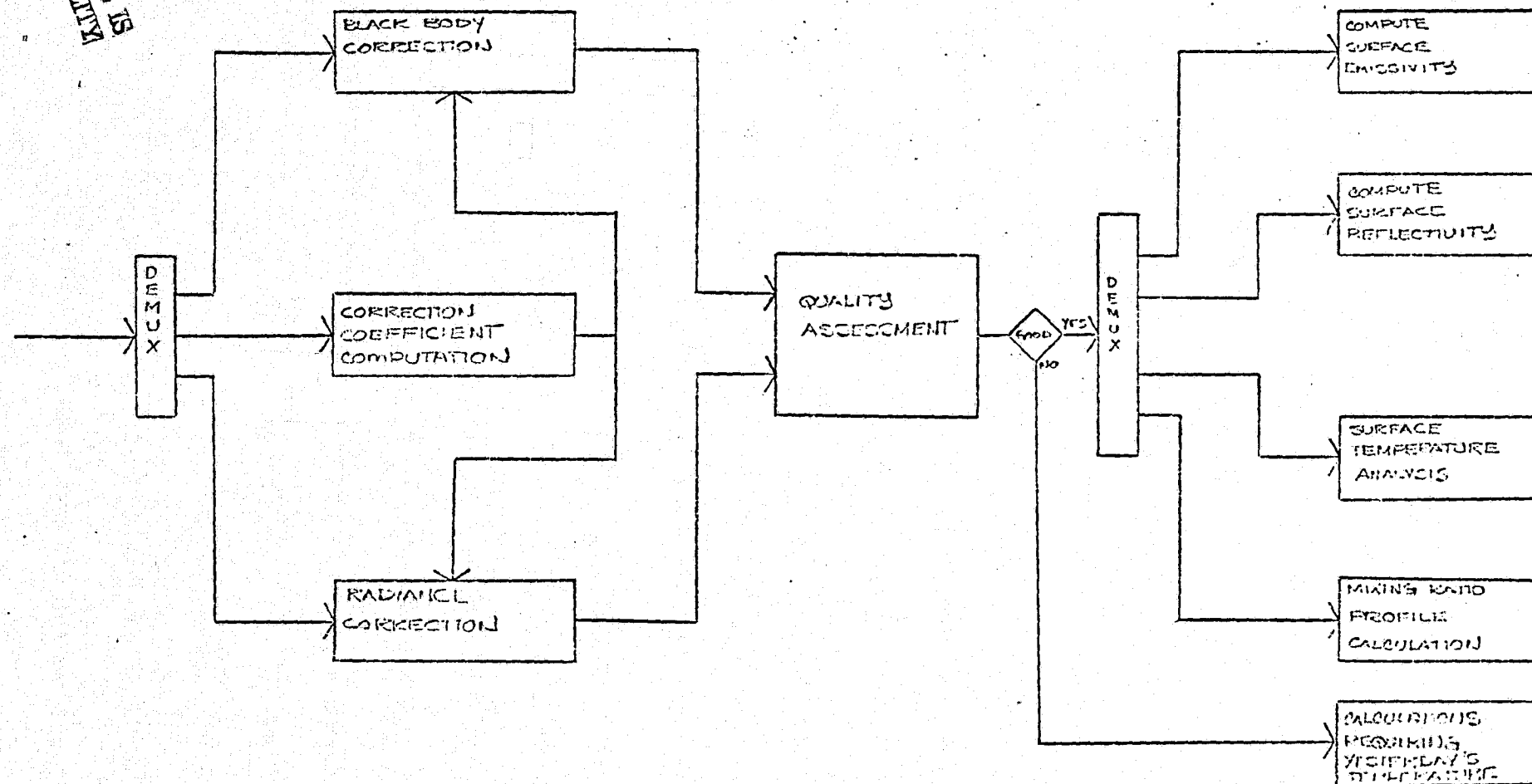


GRID FORMAT  
FIGURE 3.2-2

ORIGINAL PAGE IS  
 OF POOR QUALITY

ORIGINAL PAGE IS  
OF POOR QUALITY

3-18



SYSTEM LEVEL PROCESSING FLOW DIAGRAM

FIGURE 3.2-3

of black body correction, earth radiance correction, quality assessment, surface emissivity computations, surface reflectivity computations, surface temperature analysis, mixing ratio profile computation, and computations requiring previous day's data or data handling. Figure 3.2-4 indicates the location of the functional processes within these categories.

CATEGORY	PROCESS ASSIGNMENT
EARTH RADIANCE CORRECTION	14, 15, 16, 17, 18, 19, 23, 24, 25,
BLACK BODY CORRECTION	3, 4, 5, 6, 7, 8, 13, 19, 20, 21, 22, 26, 27, 28, 29, 30, 31, 32 33, 34, 35,
COEFFICIENT COMPUTATION	9, 10, 11, 12, 13
QUALITY ASSESSMENT	36, 37, 38, 41, 42, 44, 46, 43, 50C, 53, 59
CONTROL	1, 2, 39, 40, 43, 47, 45, 49, 50, 51, 52, 55, 54, 56, 58, 61, 62 66, 70, 86, 92, 93, 94
SURFACE EMISSIVITY	57
SURFACE REFLECTIVITY	57,
TEMPERATURE ANALYSIS	60, 63, 64, 65, 67, 68, 69, 72, 87, 88, 89, 90, 91 76, 77, 78, 79, 80
MIXING RATIO PROFILE	81, 82, 83, 84, 85
YESTERDAY'S TEMPERATURE	95

FIGURE 3.2-4

ORIGINAL PAGE IS  
OF POOR QUALITY

Initially, the output of the sensor is digitized so that a reference for conversion is required. Since the reference varies with physical conditions as well as the inherent non-linearities in each detector, the received signal must be calibrated or corrected. The scan line contains calibration data which is used in the following manner: For a given measured analog value, the theoretical digital value may be defined analytically based on the characteristics of the converter; therefore, a calibration reference is required to re-compute the theoretical or estimated relationship. Processes for the coefficient computation shown in Figure 3.2-4 are used to recompute the calibration curve.

Every sixteen scan lines, thirty-two calibration values are obtained from a reference source. Every sixteen scan lines, the reference is reset and the process repeated. Since the measured value corresponds directly to an apriori known theoretical value, a new curve may be plotted. Any sensed value must fall on the curve and the measured signal must be located on this curve. The correction coefficients reduce to a gain and an offset based on the calibration data. These coefficients are stored in a memory accessed as a function of the sensed values. This process may be implemented by either of two techniques. The first is to store a complete set of 32 points and recompute the entire curve every sixteen scan lines. This technique maintains a low frequency but causes abrupt changes in the calibration curve and reduces the continuity between adjacent scan lines at the curve switching point. In addition, any noise present will have a dominant effect. Further, the technique requires a bi-phase memory.

The second technique is to compute and update the calibration curve on a scan line basis. Although the frequency of update is sixteen times that of the first technique there ~~are~~ no abrupt changes in the coefficient table.

Consequently, continuity is maintained with an inherent smoothing. This technique requires a single phase memory which may be easily updated due to the system timing. From examination of the overflow, there is no read-while-write requirement. Therefore, the second approach, i.e. scan line rate update, has been selected.

Having established the calibration coefficients, each sensed value must be corrected. The process is identical for the earth sensed and black body values with additional preprocessing for the black body. In measuring the black bodies (BB) a scan line of 42 samples per BB is received. Each value will differ slightly due to the instantaneous properties of the electronics and variations in the BB as shown in Figure 3.2-5.

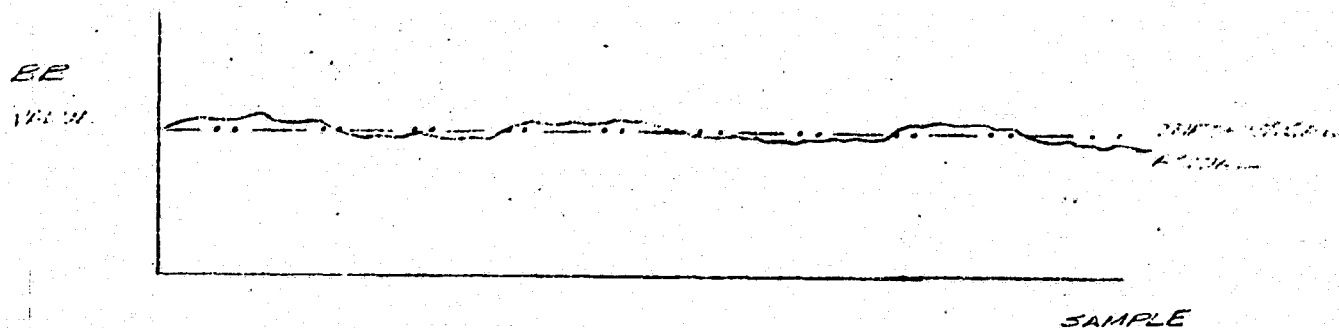


FIGURE 3.2-5

Although the deviations are small, the selection of a discrete value results in a maximum possible error. To minimize the error, the average value for the scan line is used. The signal conditioning is process 3 resulting in the uniform distribution shown in Figure 3.2-6.

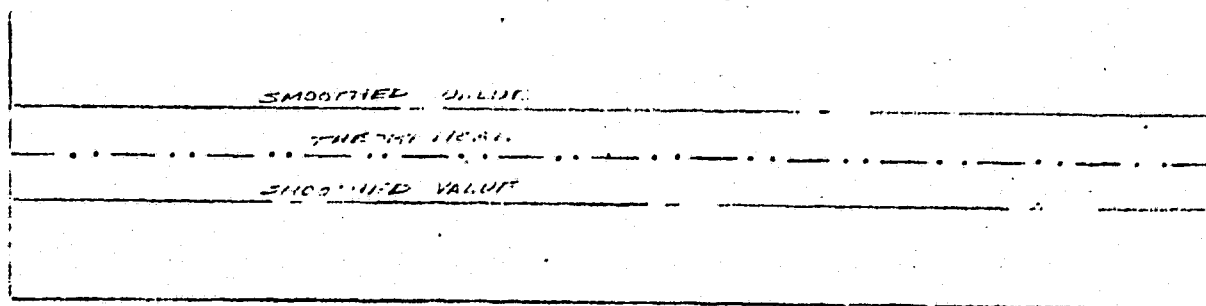


FIGURE 3.2-6

SMOOTHED

The second black body unique process is to average the BB values obtained before and after the grid data. This is process 4; from process 4 on, the correction process for the black body and the earth values is identical. These functional processes require the processes listed in Figure 3.2-5. In order to correct each word, the following procedure is required; Based on the magnitude of the received value, a gain and offset are selected and a new value computed which is the value of the signal plus non-linearity. By subtracting the theoretical value, the deviation present in the measured signal is computed. Having determined the deviation, the received signal is corrected by algebraically adding the offset. It is noted that the correction of these data words may be actually performed by the same device due to the location of the black body data and the earth data in the grid.

Prior to any earth data processing a quality assessment and normalization process is performed. Since the black body is a temperature, and is related to radiance by a planck function some preprocessing is required. Three black body values are obtained and a curve plotted in the same manner that the electronic calibration was computed. The resultant curve is the corrected temperature. Since the

detectors are spectrally separated the effective temperature must be computed for the specific detector in question and converted to radiance as shown in processes 21 through 35.

Figure 3.2-7 shows the geometry of the sensor/earth relationship.

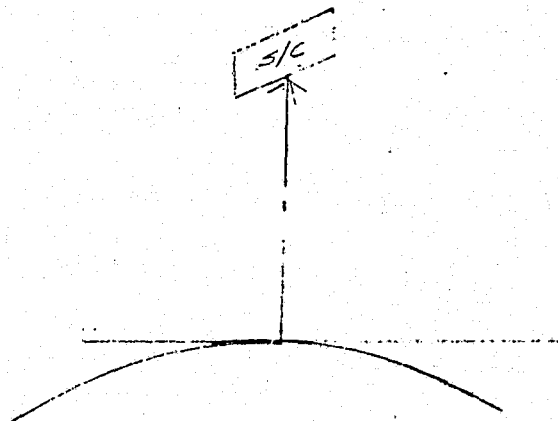


FIGURE 3.2-7

Because of the different distances traveled as a function of the scan angle, the radiant energy is diffused through a longer path so that the data must be normalized about some point. The standard technique for spacecraft is to normalize about the nadir. The normalized data is assessed with respect to its quality. This process is comprised of the functions listed in Figure 3.2-5 with the parameters determined from user evaluation models. The major aspect of the quality assessment establishes the requirement to use the measured data or the previous day's data in its place, as well as the requirement to compute emissivity and reflectivity or to use assumed values.

The original process substitutes the previous day's temperature for any point which does not satisfy the acceptance criteria defined in Figure 3.2-3. This approach is not feasible for an onboard process because it requires large memories which must be updated on a daily basis. The rationale behind the substitution is that a valid temperature cannot be measured when the target is cloud covered; the use of yesterday's temperature is an approximation based on the fact that it is closer to the actual temperature than the measurement attempted through clouds. An equally valid approximation may be made by averaging the four nearest surrounding clear neighbors. Support for this approach is provided by the Temperature Analysis processing which determines the validity of the temperature at a point by comparing it to its neighbors. The point must not deviate from the average of its neighbors by more than a preset amount.

This process is readily implementable onboard. Further, a flag will be set when this process is implemented so that further processing on the ground can replace it with yesterday's temperature if this is deemed more desirable.

#### Hardware/Software/Firmware Trade-Off

The preliminary implementation must be analyzed with respect to groups of functions. The low frequency of the sensor eliminates the processing constraints imposed by a sensor such as the ATS. Consequently, the major criteria for implementation are the economics of the design. There is no justification for implementation in hardware, other than the memory and delay processes.

The implementation trade-offs are between firmware and software. Like hardware, firmware requires some degree of random logic design which, with respect to the IRS, closely resembles the Wilkes Processor with an additional external register capability. The prime software consideration is the micro processor. The process



requires a vast distribution of comparisons and resultant decisions so that the firmware approach is rather complex. The capability is inherent in a micro-processor and is easily implemented with a six register machine. Therefore, the approach is selected and is to be implemented in a multiple CPU micro-computer augmented with a firmware fast arithmetic and a hardware comparator. Since the two augmenting units are external it is recommended that the micro-processor used have the characteristic of treating the peripherals as a memory location. The primary reasons for this approach are the low data rate, the decision making requirements, and economics.

# IRS DATA PROCESSING

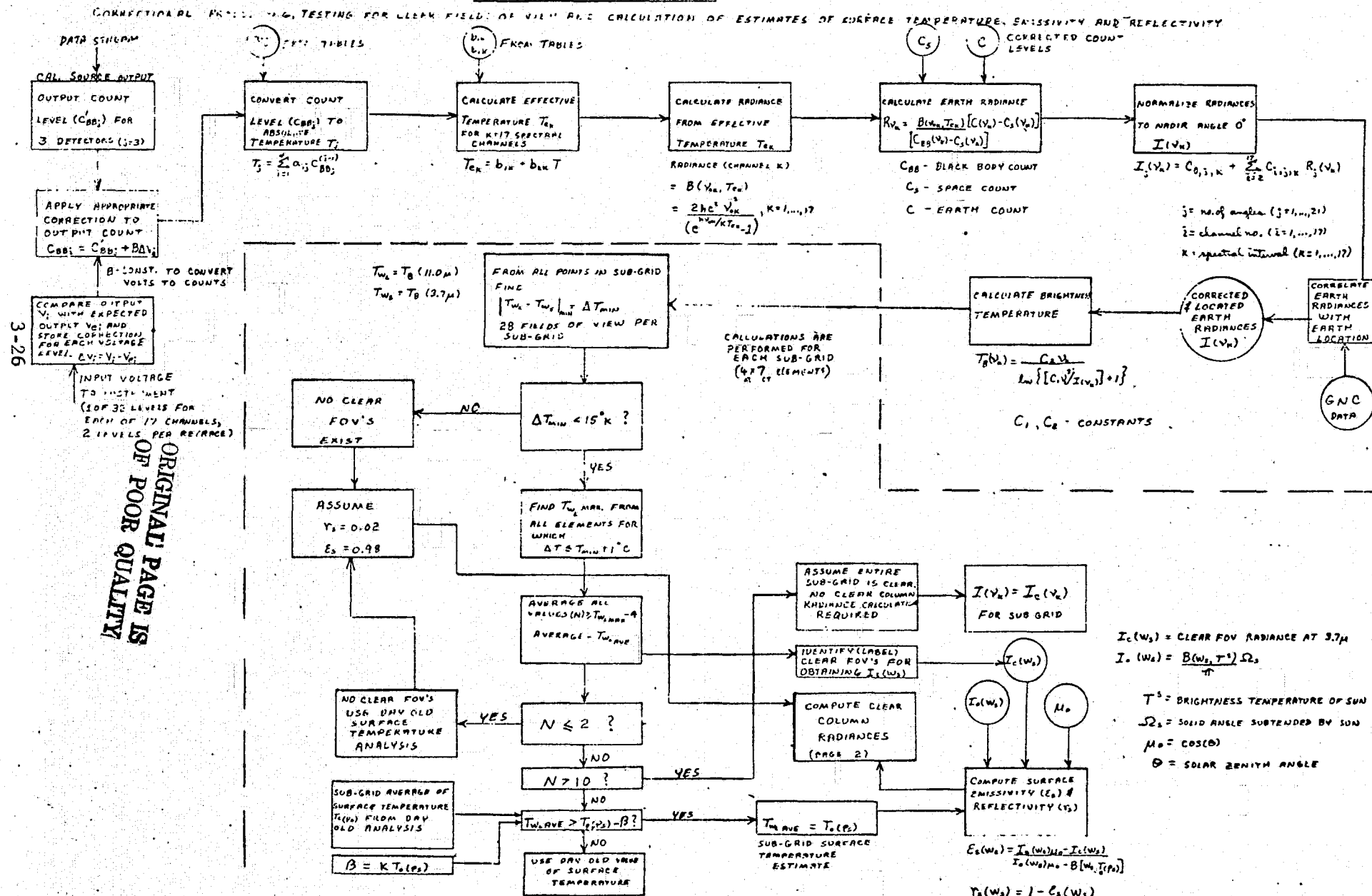


Figure 3.2-8a

# IRS DATA PRESENTING

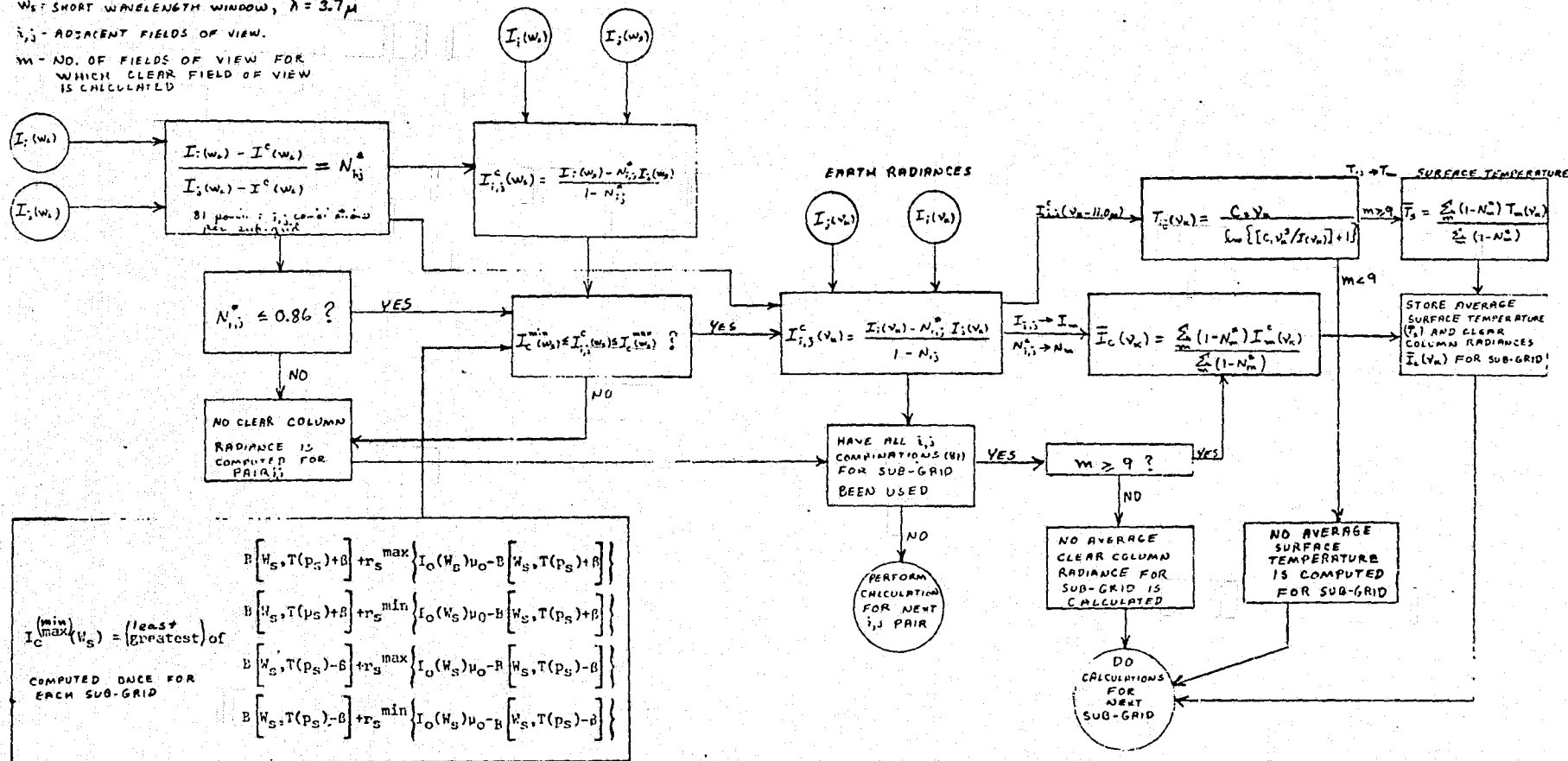
## CALCULATION OF SURFACE TEMPERATURE AND CLEAR COLUMN RADIANCE FOR SUB-GRID

$W_L$  - LONG WAVELENGTH WINDOW,  $\lambda = 11.0 \mu$

$W_S$  - SHORT WAVELENGTH WINDOW,  $\lambda = 3.7 \mu$

$i, j$  - ADJACENT FIELDS OF VIEW

$m$  - NO. OF FIELDS OF VIEW FOR WHICH CLEAR FIELD OF VIEW IS CALCULATED



$B[W_S, T(p_s) + B]$  - PLANCK FUNCTION (SEE PAGE 1)

$T(p_s)$  - SURFACE TEMPERATURE ESTIMATE (FROM PAGE 1)

$B = K T(p_s)$  WHERE  $K$  IS A PREDETERMINED CONSTANT

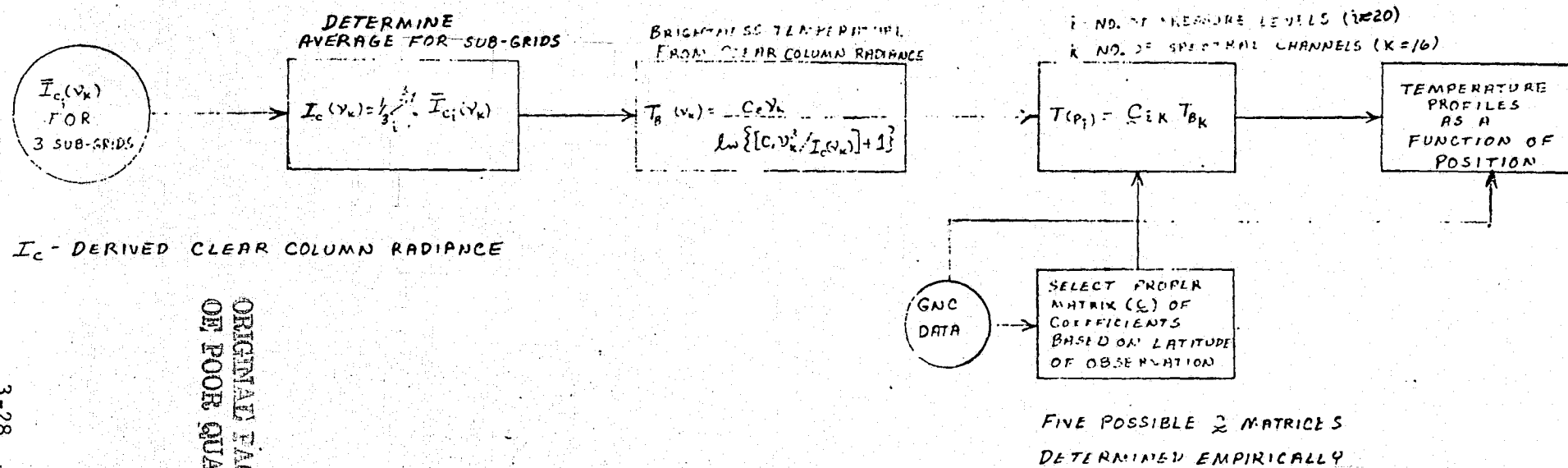
$I_0(W_S), \mu_0$  - SEE PAGE ONE

$Y_s^{max} = Y_s + 0.10 Y_s$  WHERE  $Y_s$  IS OBTAINED FROM PAGE ONE

$Y_s^{min} = 0$

Figure 3.2-8b

# TEMPERATURE PROFILE CALCULATION



## MIXING RATIO PROFILE CALCULATION

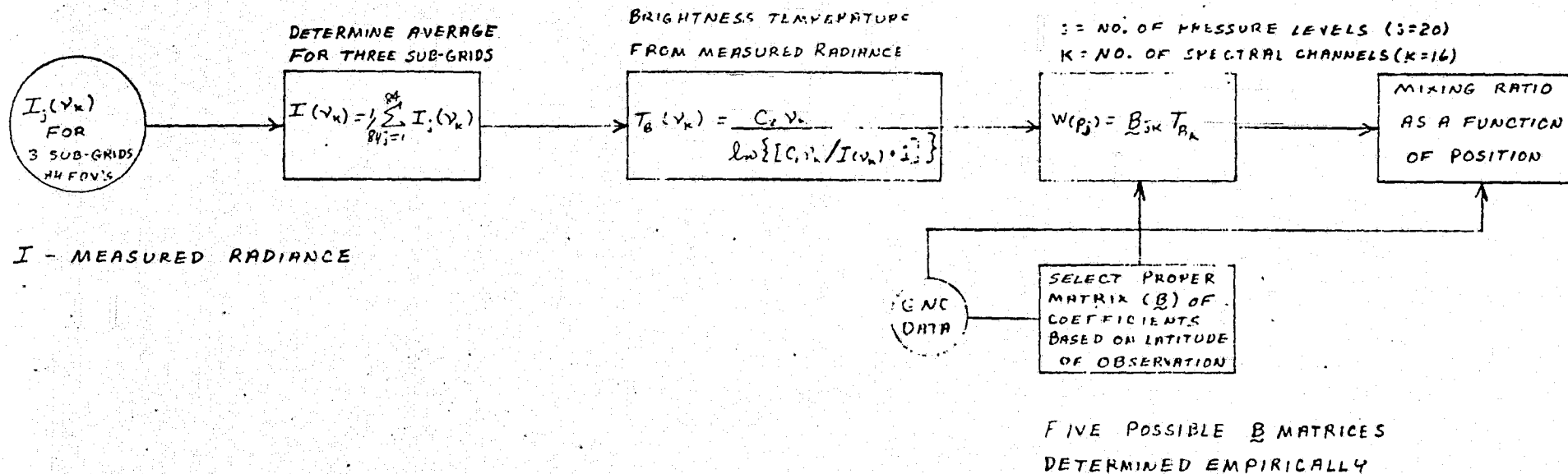


Figure 3.2-8c

ORIGINAL PAGE IS  
OF POOR QUALITY

# SURFACE TEMPERATURE ANALYSIS

4

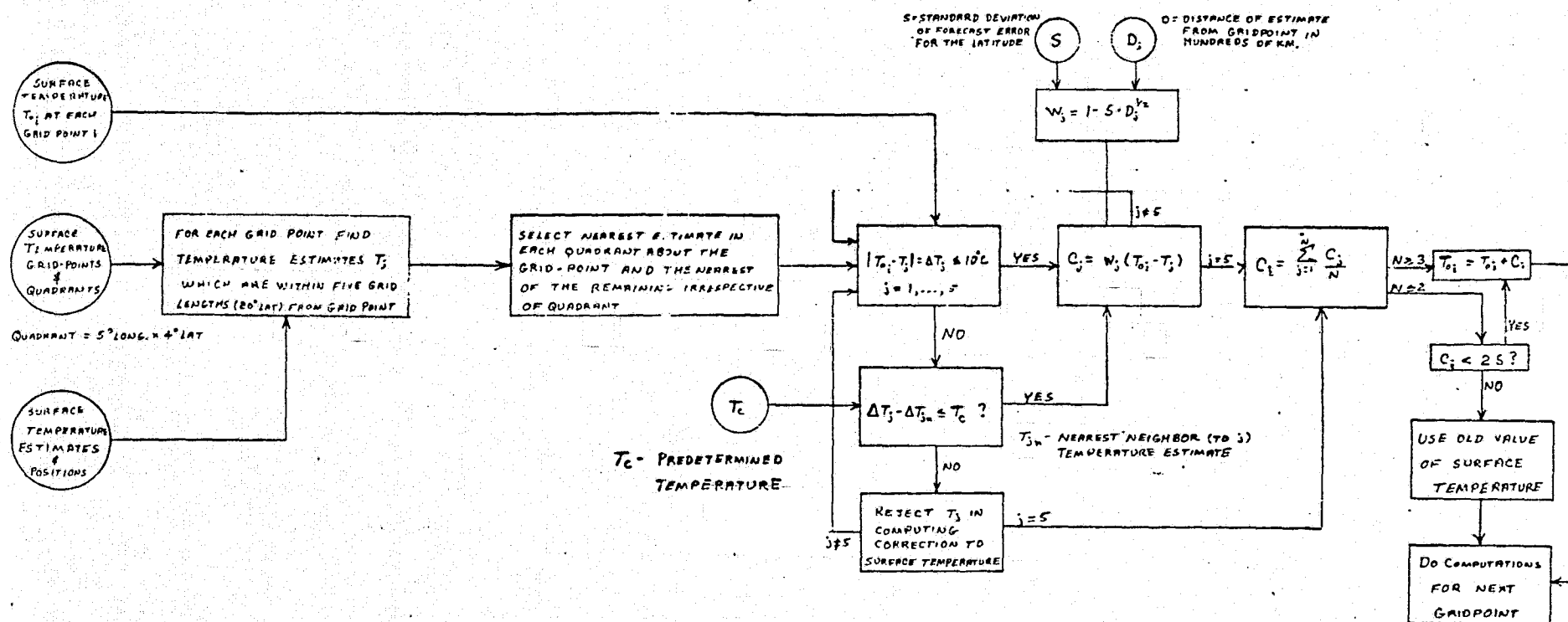
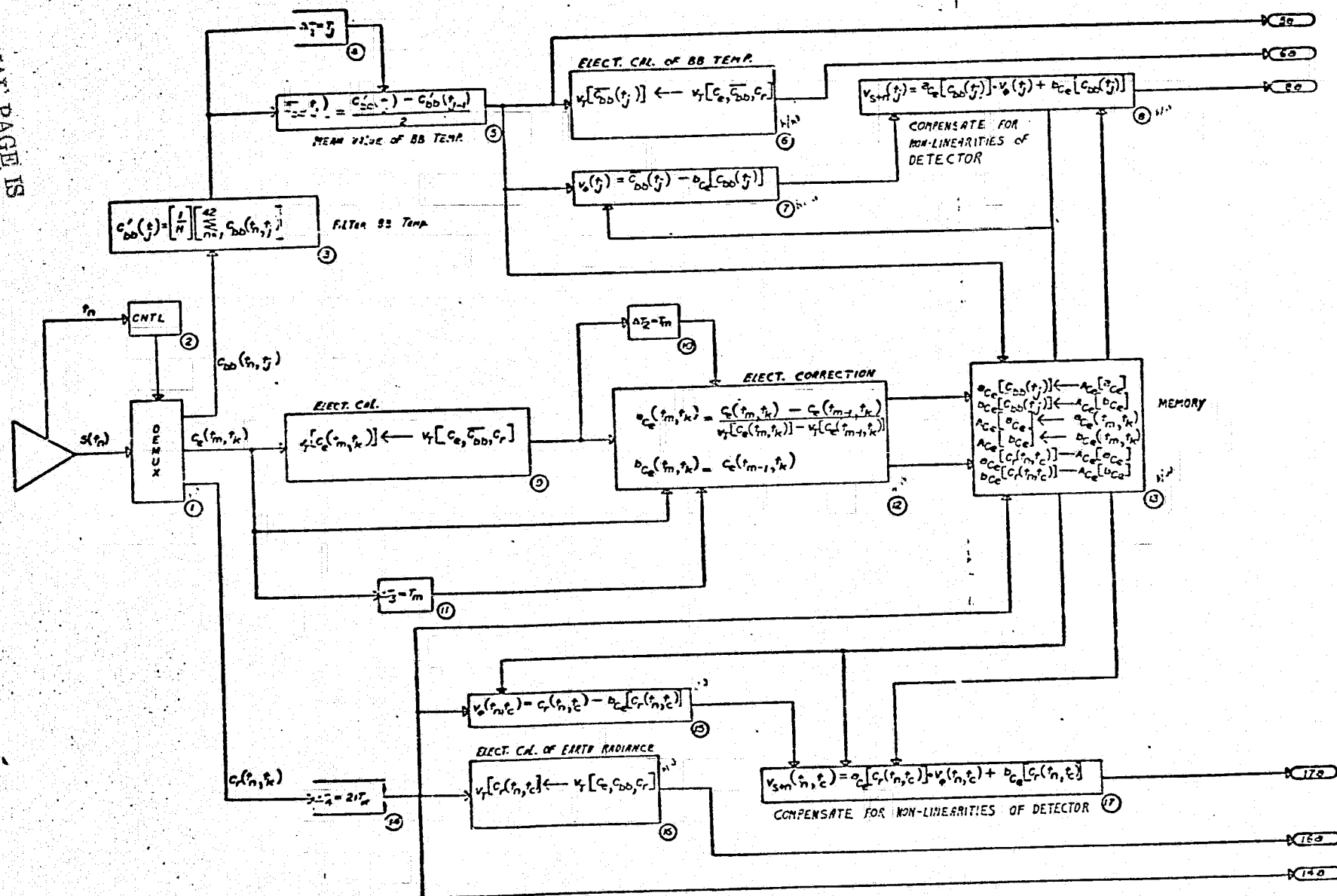


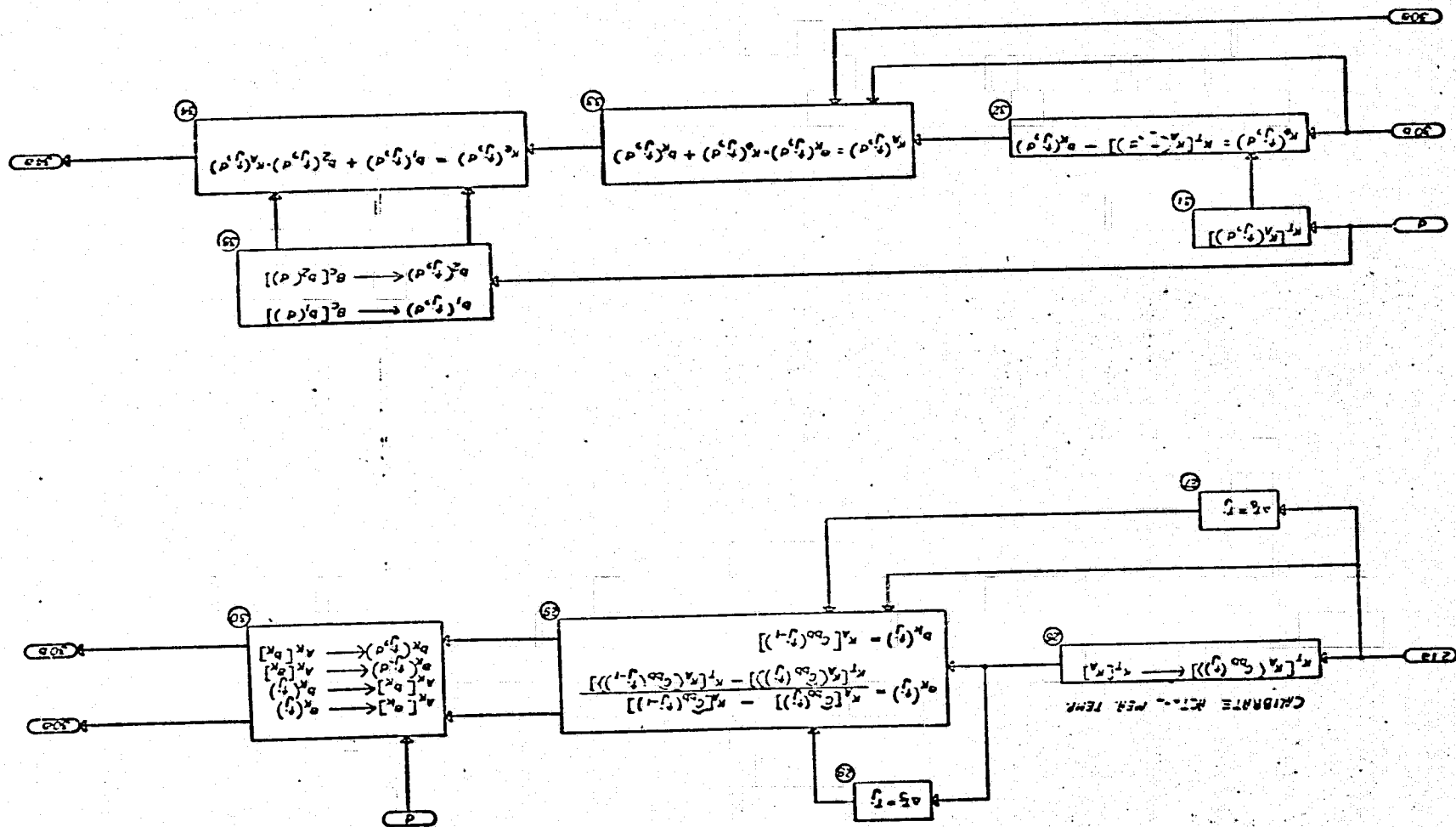
Figure 3.2-8d

ORIGINAL PAGE IS  
OF POOR QUALITY

# INFRARED SPECTROMETER FUNCTION FLOW DIAGRAM

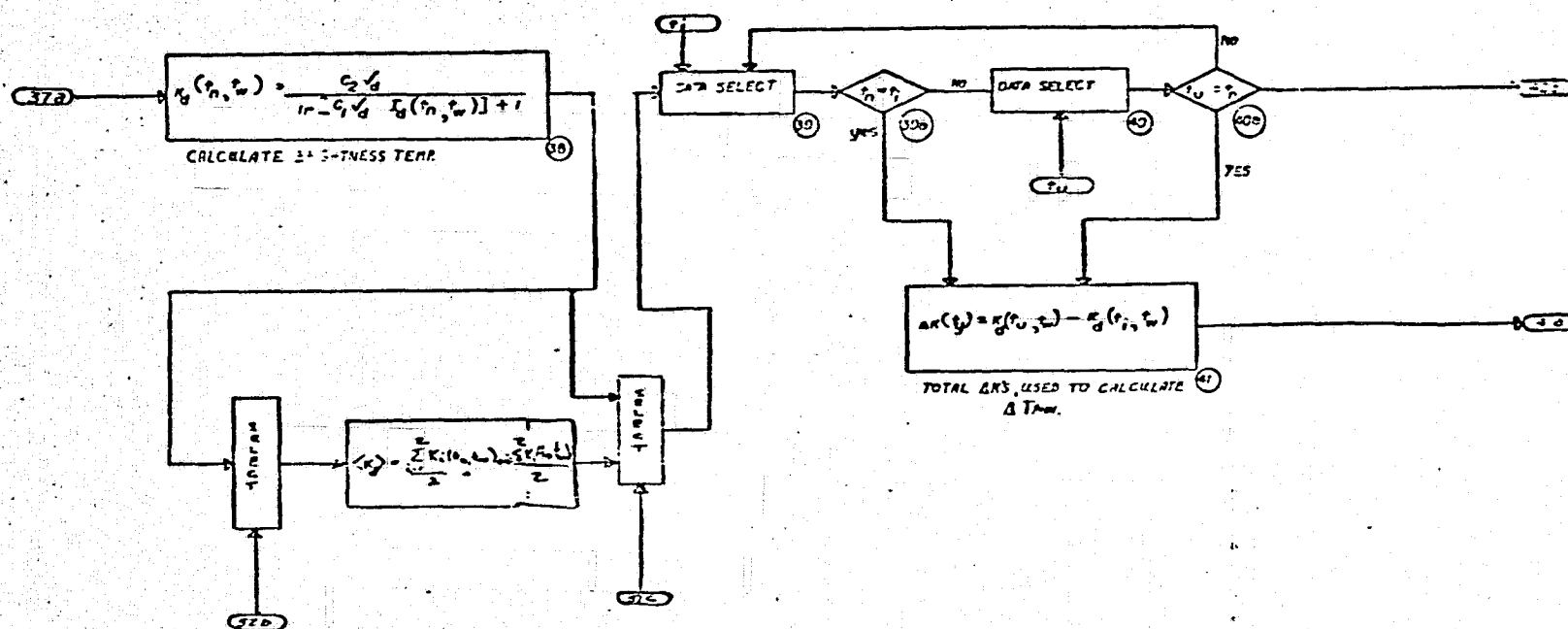
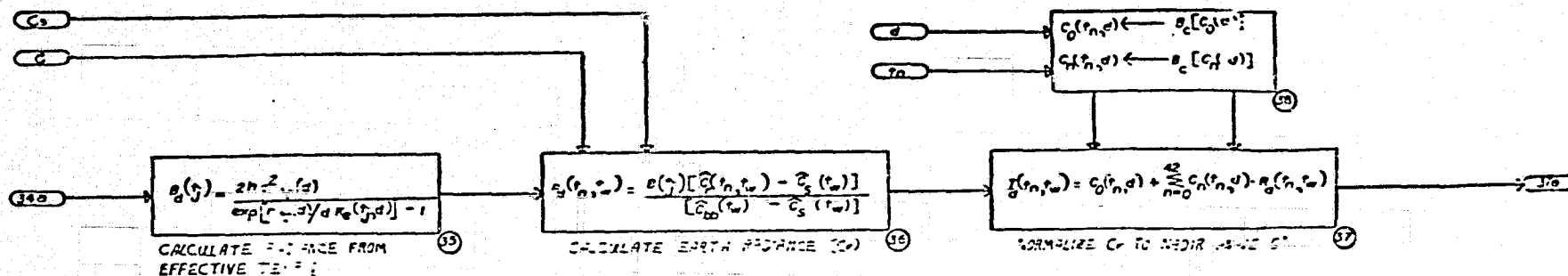




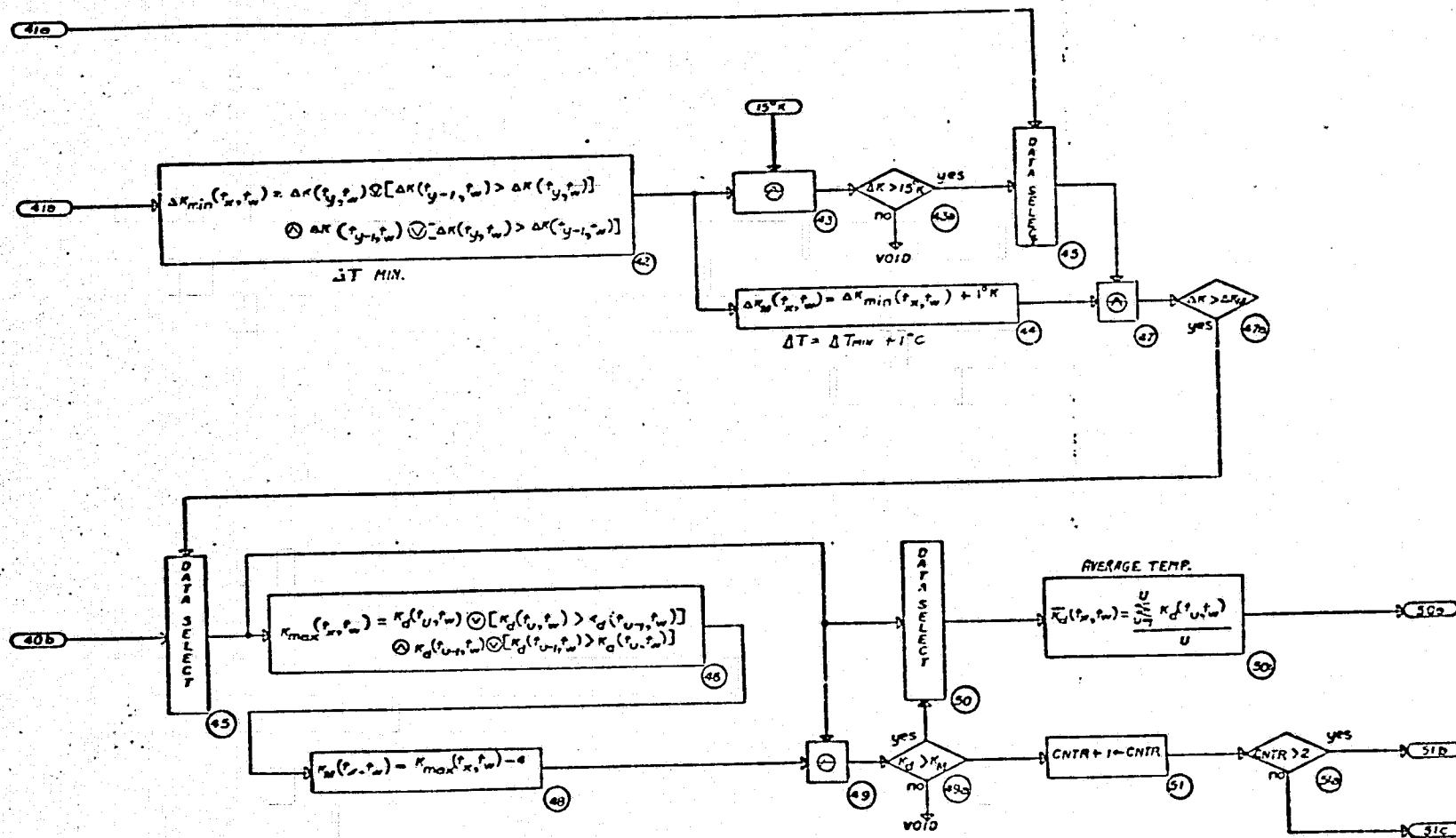


ORIGINAL PAGE IS  
OF POOR QUALITY



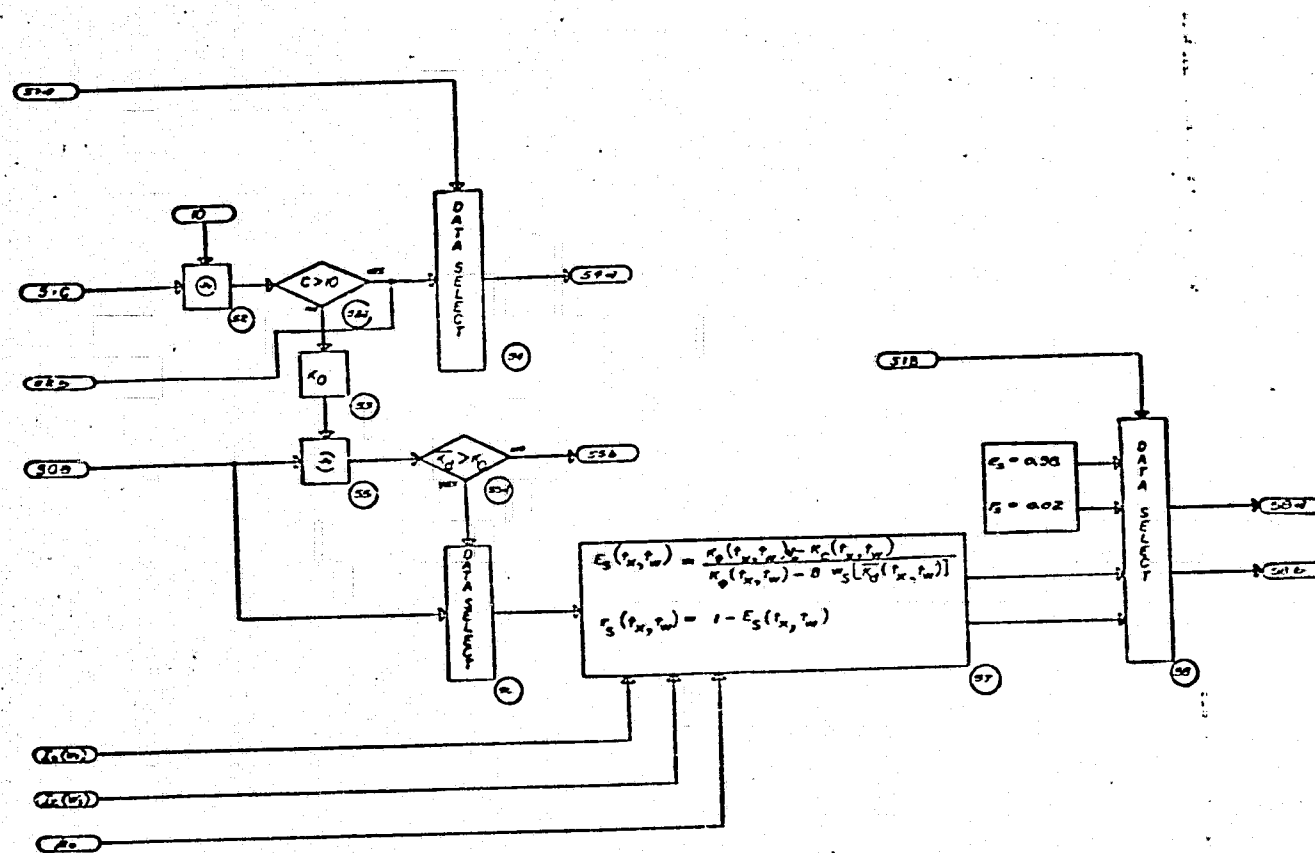


ORIGINAL PAGE IS  
OF POOR QUALITY



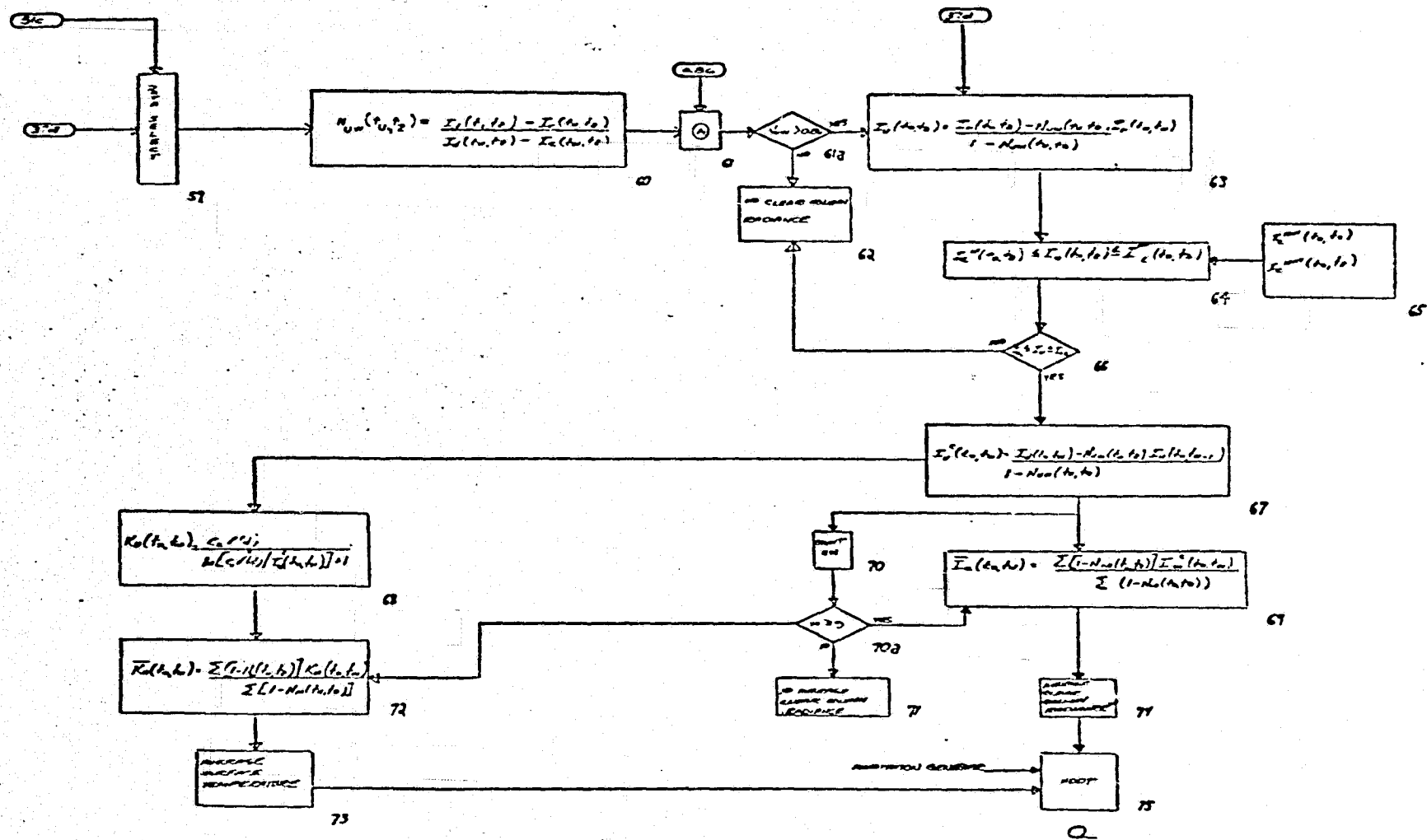
DETECTOR TEMPERATURE AVERAGER

# CALCULATION OF SURFACE TEMPERATURE AND CLEAR COLUMN RADIANCE FOR SUB-GRID

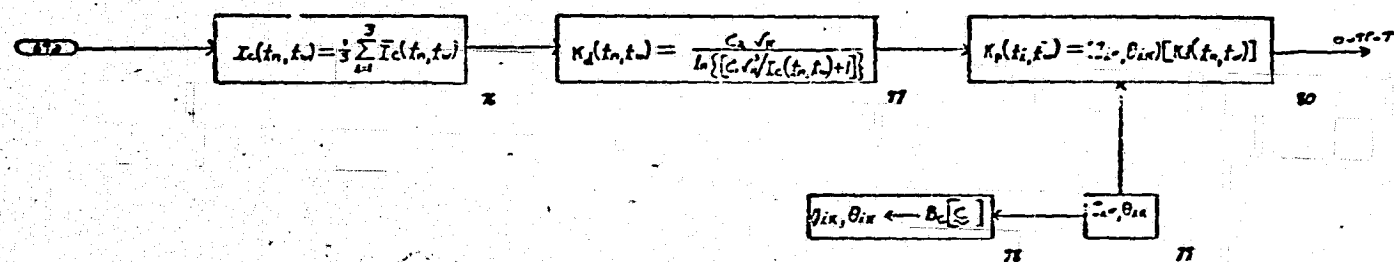


COMPUTE SURFACE EMISSIVITY AND REFLECTIVITY

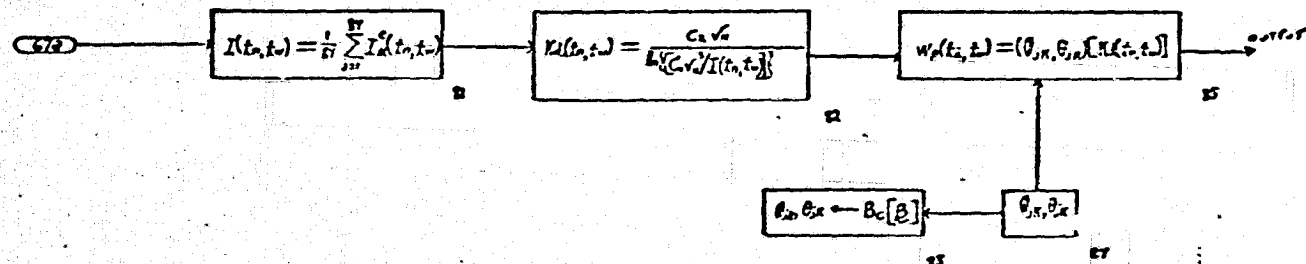
ORIGINAL PAGE IS  
OF POOR QUALITY



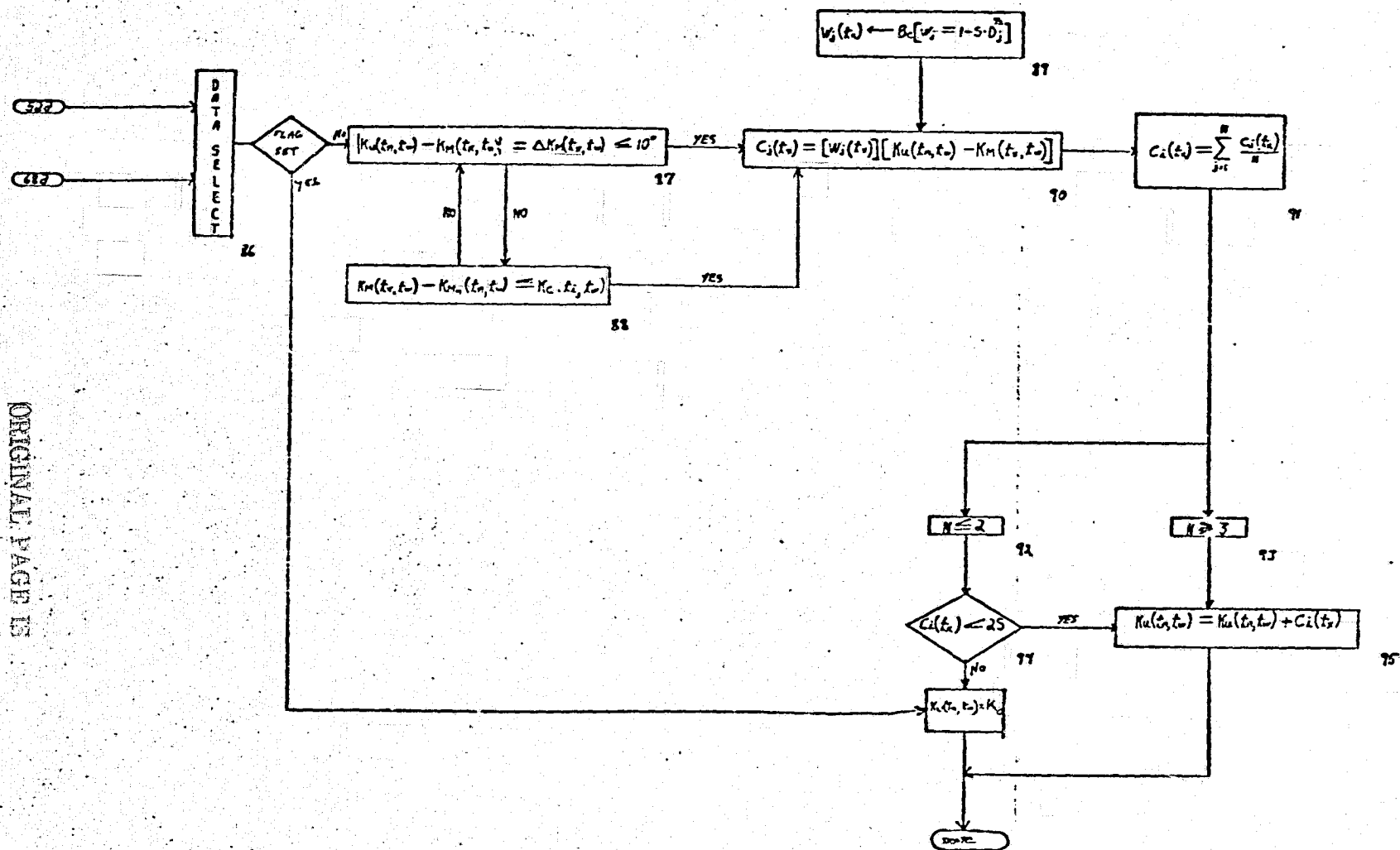
# TEMPERATURE PROFILE CALCULATION



# MIXING RATIO PROFILE CALCULATION



ORIGINAL PAGE IS  
OF POOR QUALITY



SURFACE TEMPERATURE ANALYSIS

### 3.3 RADIOMETER/SCATTEROMETER (RADSCAT)

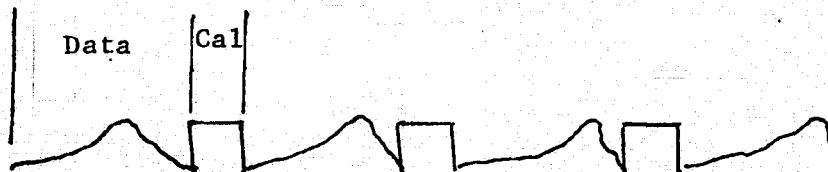
This section describes the functional flow developed for the Radscat and the rationale for the various trade-offs effected.

#### Introduction

The radiometer scatterometer is a microwave instrument developed for the measurement of physical science phenomena. The radiometer portion of the sensor is a stable receiver which measures antenna noise power within a finite bandwidth. The noise power measured is directly proportional to the apparent antenna temperature allowing thermal emissivity to be computed. The apparent antenna temperature is related to the area brightness temperature by the antenna gain function and properties of the target.

The scatterometer is an instrument developed to measure the amount of energy backscattered from a target. This instrument is an active instrument i.e. its operation requires the transmission of a signal and the measurement of its reflection. A more detailed description is contained in appendix D of the Onboard Experiment Data Support Facility Task I Report NAS Contract 9-14651 dated September 1975.

The sensor generates a multiplexed multi-channel output shown in Figure 3.3-1.



RADSCAT DATA FORMAT

FIGURE 3.3-1

The output signal is comprised of the measured data and housekeeping and calibration data. The calibration data is the spacecraft attitude synchronized to spacecraft time. The highest data rates correspond to a 1.876 millisecond word period and an 18.76 millisecond frame period. The sensor data precedes its calibration data in time so that a re-formatting is required in the same fashion as for the ATS. The output data must be referenced to the target location on the earth's surface based on GNC data, the significant aspect of the RADSCAT is the amount of ancillary data required to process the primary sensor data.

#### INFORMATION PROCESSING

The processing from a real time aspect is discussed based on the procedural algorithms shown in the functional flow diagrams (Figure 3.3-2). The basic functions are extensive but realizable because of the relatively low data rate. The processes include some complex functions such as matrix multiplications but are dominated by trigonometric and inverse trigonometric functions.

Initially, the data must be corrected or normalized to remove integration drifts and offsets in process 1. The correction process is dependent on the data word as a function of time and selected constants determined apriori. Based on the corrected values the power ratio of returned to transmitted is computed as shown in process 2. The time relationships of the parameters simplify the actual process. Based on the power ratio or back scattering  $\sigma_0$  is computed as a function of time



in process 3. The backscattered energy is corrected, i.e. normalized in process 4 for incidence angle, referenced from a spacecraft coordinate system to an earth projection system. Although a specific coordinate system is used, the data may be converted to any standard projection through existing transforms. The functions in processes 5, 6, 7, 8, 9, and 10 are relatively straight forward except for the matrix multiplication. It is important to note that regardless of the implementation, the matrix multiplication with double precision presents a formidable implementation problem. Based on state-of-the-art hardware and software, the multiplications shown in processes 6 and 7 require unique techniques because the coefficients are complex trigonometric relationships that are time varying at the frame rate. The remaining normalization processes are based on geodetics and are identical to those described in Section 3.1.

The second requirement is to process the antenna noise power to determine the apparent antenna power. This is performed by processes 11 and 12. The conversion from noise power is based on emissivity, i.e. treating the source as a black body, and the knowledge of the receiver. The process is a straightforward algorithm for a finite bandwidth.

The significant processing requirements are the matrix multiplication, trigonometric functions, and dot product. From a functional flow aspect the RAD/SCAT possesses no inherent problems for total spaceborne processing but its effectiveness is dependent on the processing capability of the composite OEDSF processor.

#### HARDWARE/FIRMWARE/SOFTWARE Trade-off

Based on the low frequency data rates and simple processing requirements, a hardware implementation is neither warranted nor justified. A pure software approach is neither practical nor feasible due to the trigonometric, matrix multiply, and dot product functions. The matrix multiply provides the partitioning basis. If a microprocessor is used a 3 x 3 matrix multiply requires 8.37 milliseconds per matrix not including overhead. A study performed by General Electric on microprocessors

(DCS III Attitude Transformations for skewed wheels by R.A. Sergio April 1975)

has shown that the conventional 8 bit microprocessor requires an additional 70 percent overhead, so that a matrix multiply required 14.23 milliseconds. Since the process must be completed within 18.76 milliseconds and repeated four times, a pure software approach is impossible. The referenced report also shows that a microprocessor augmented with fast arithmetic requires 50 microseconds per multiply and 2.0 microseconds per add with each coefficient requiring four multiplies and ten additions, so that a major portion of the duty cycle is required. This same rationale may be used for the dot product and trigonometric functions. These functions will be implemented in firmware with microprocessors.

The remaining functions can easily be implemented in software with conventional 8 bit microprocessors, however the microprocessor required by the aforementioned processes can readily be augmented with hardware multiply and divide. This augmentation reduces the multiply time from approximately 250  $\mu$ s to 50  $\mu$ s or 3.5  $\mu$ s depending on the CPU and architecture. (An Intel 8080 requires 50  $\mu$ s while a Motorola 6800 requires 3.5  $\mu$ s). Note that the actual CPU must be selected on the basis of a trade-off between 8 bit versus 16 bit machines because the sensor outputs 10 bit words.

RADSCAT

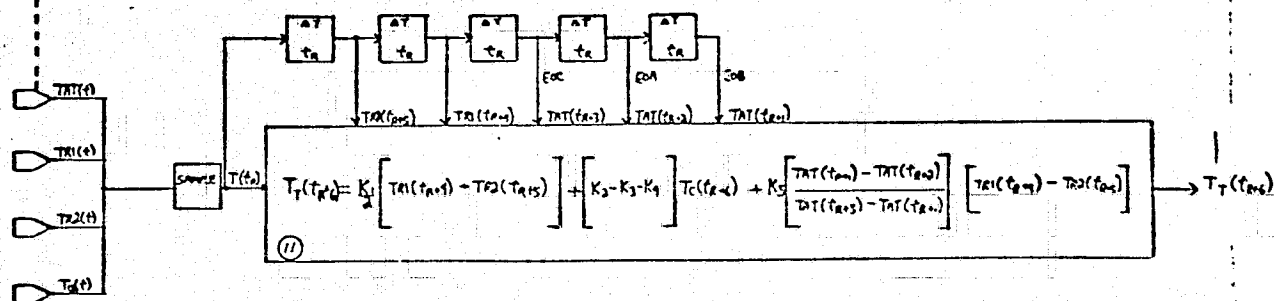
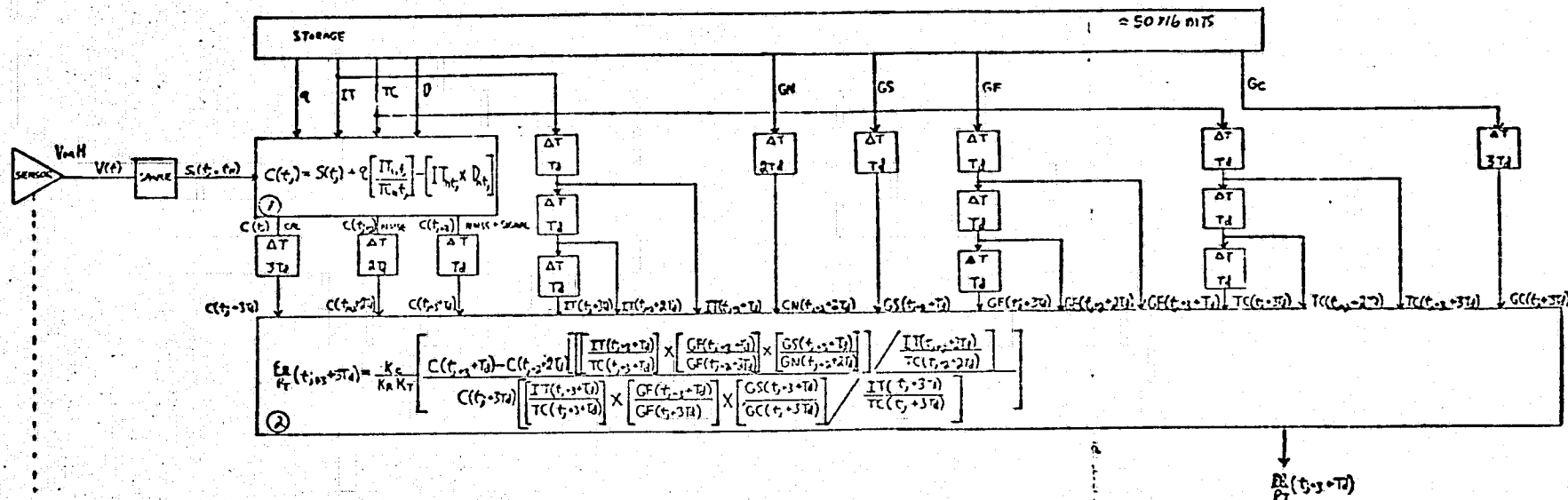
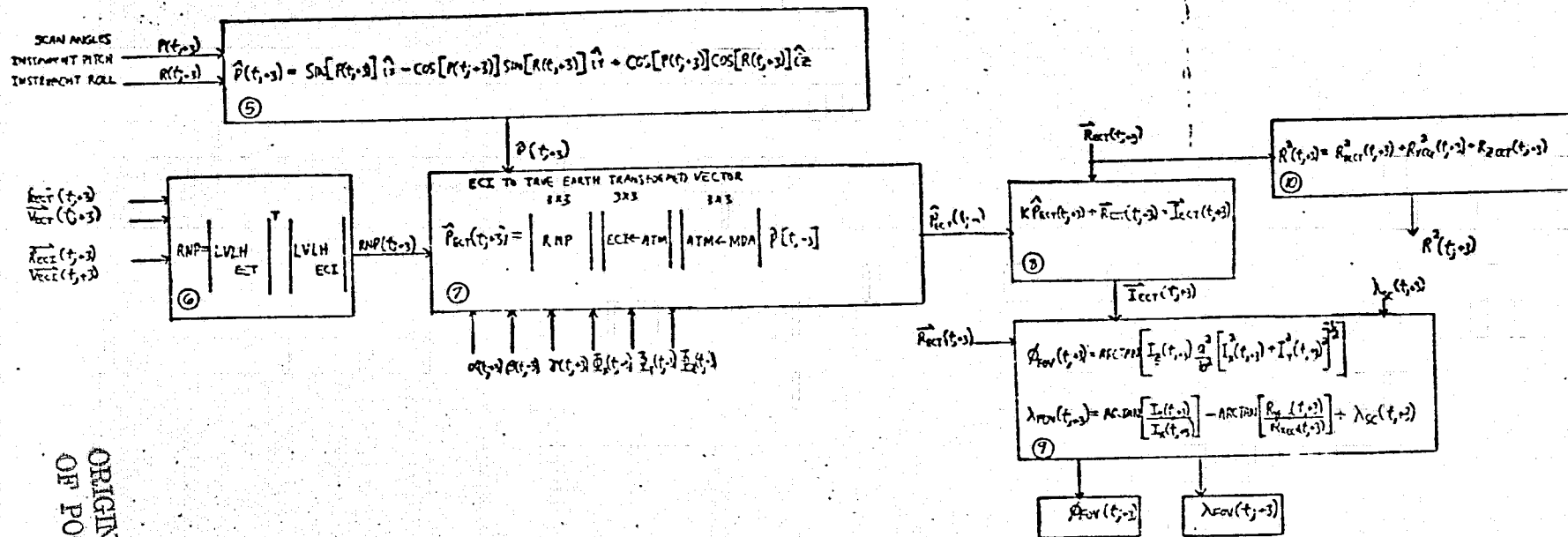


Figure 3.3-2a

# RADSCAT



ORIGINAL PAGE IS  
OF POOR QUALITY

$$\begin{aligned}
 & \text{LVH ECI} \Rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \\ 3 & 2 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} \hat{x}_{ECI} & \hat{y}_{ECI} & \hat{z}_{ECI} \\ \hat{y}_{ECI} & \hat{x}_{ECI} & \hat{z}_{ECI} \\ \hat{z}_{ECI} & \hat{y}_{ECI} & \hat{x}_{ECI} \end{bmatrix} \\
 & \text{LVH ECI} \Rightarrow \text{IDENTICAL EXCEPT FOR INVERSE UNIT VECTORS AND ECI VECTOR COEF.}
 \end{aligned}$$

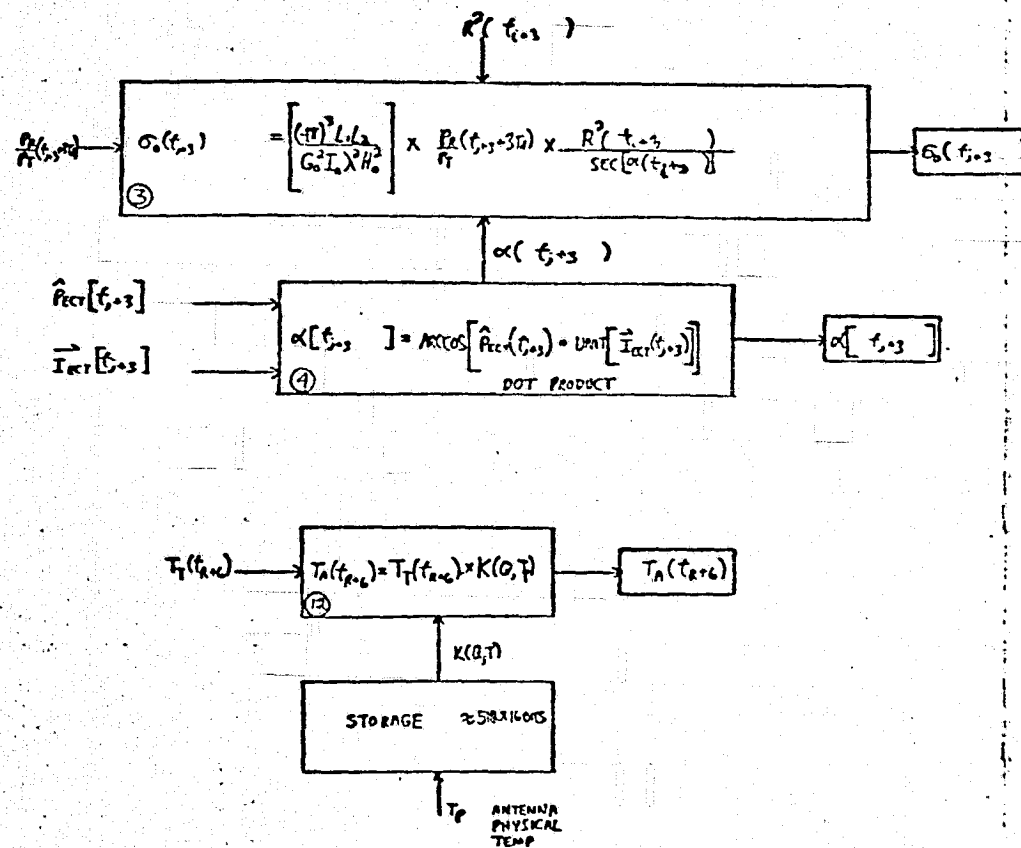
$$\begin{aligned}
 & \text{ECI} \leftarrow \text{ATM} \Rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \\ 3 & 2 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} \hat{x}_{ECI} & \hat{y}_{ECI} & \hat{z}_{ECI} \\ \hat{y}_{ECI} & \hat{x}_{ECI} & \hat{z}_{ECI} \\ \hat{z}_{ECI} & \hat{y}_{ECI} & \hat{x}_{ECI} \end{bmatrix} \\
 & \text{ECI} \leftarrow \text{ATM} \Rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \\ 3 & 2 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} \hat{x}_{ECI} & \hat{y}_{ECI} & \hat{z}_{ECI} \\ \hat{y}_{ECI} & \hat{x}_{ECI} & \hat{z}_{ECI} \\ \hat{z}_{ECI} & \hat{y}_{ECI} & \hat{x}_{ECI} \end{bmatrix}
 \end{aligned}$$

$$\begin{aligned}
 & \text{ATM} \leftarrow \text{MDA} \Rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \\ 3 & 2 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} \hat{x}_{ATM} & \hat{y}_{ATM} & \hat{z}_{ATM} \\ \hat{y}_{ATM} & \hat{x}_{ATM} & \hat{z}_{ATM} \\ \hat{z}_{ATM} & \hat{y}_{ATM} & \hat{x}_{ATM} \end{bmatrix} \\
 & \text{ATM} \leftarrow \text{MDA} \Rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \\ 3 & 2 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} \hat{x}_{ATM} & \hat{y}_{ATM} & \hat{z}_{ATM} \\ \hat{y}_{ATM} & \hat{x}_{ATM} & \hat{z}_{ATM} \\ \hat{z}_{ATM} & \hat{y}_{ATM} & \hat{x}_{ATM} \end{bmatrix}
 \end{aligned}$$

Figure 3.3-2b

ORIGINAL PAGE IS  
OF POOR QUALITY

# RADSCAT

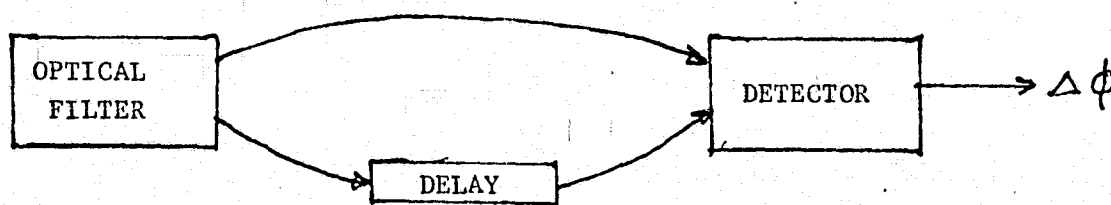


### 3.4 CIMATS

This section describes the functional flow diagram of the CIMATS and the rationale for the various tradeoffs effected.

#### INTRODUCTION

CIMATS is an instrument developed to measure the atmospheric concentration of trace gases by interferometric correlation. A complete description of the instrument and its operation is contained in Appendix C of the OEDSF Task 1 report. Interferometric correlation is based on an optical modulation using a frequency dependent delay line. A received optical domain signal is converged on the same detector after being transmitted along two separate paths as shown in figure 3.4-1.



INTERFEROMETRIC APPROACH  
Figure 3.4-1

One transmission path contains the frequency dependent delay line so that the output of the detector is a phase difference. A set of these is termed an interferogram. A simplified interferogram generation is shown in Figure 3.4-2. An interferogram is similar in appearance and use to a spectrogram. Since the delay is dependent on the frequency of the radiant energy that is incident on the aperture, the interferogram is uniquely defined for a composite gas concentration.

The determination of the actual concentration is achieved by extracting the frequency component of the desired gas. This filter may be implemented by a variety of techniques. The present technique is to use frequency differencing.

The extraction process is achieved in the following manner:

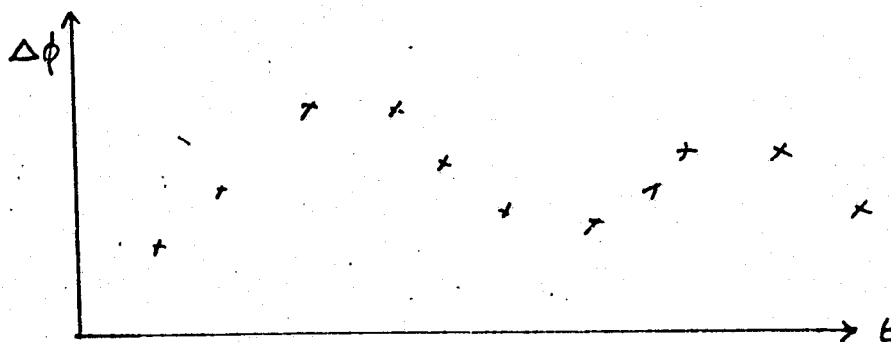


Figure 3.4-2

The measured interferogram is compared or subtracted from a calibrated reference. Since the residue is the concentration of the remaining trace gases the concentration may be computed directly. Obviously, this signal subtraction is viable only in narrow band systems with an apriori knowledge of the area of interest. The spectral regions of interest are thermal and non-thermal areas.

Interferograms produced in the non-thermal region are relatively independent of temperature profiles so that thermal corrections are not required. In the thermal mode, particularly in measurements about the nadir, the data exhibits a strong dependence on temperature requiring an additional correction process.

### PROCESSING

The fore-optics result in a  $7^\circ$  field of view which requires approximately 1 second delay in the scan range. In addition, the sensor is designed to operate in two frequency bands i.e. one thermal out of five possible and one non-thermal out of five possible creating one species map.

There are two mapping modes available: Nadir measurements and limb measurements. The nadir measurements point the instrument toward the ground along the vehicle local vertical. The limb measurement, which can be made only once per orbit, points the instrument at the sun through the atmosphere, obtaining measured values of constituent concentration as a function of altitude.

These geometries are shown in figure 3.4-3:

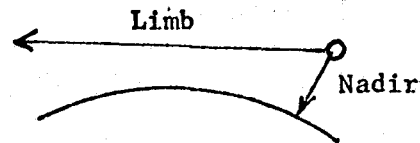


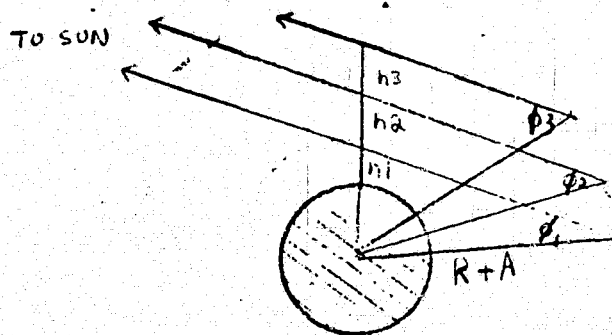
Figure 3.4-3

Figure 3.4-5 is a process requirements diagram for the CIMATS. Figure 3.4-6 is the functional flow diagram converting the required processes to real-time.

Initially, the data is sampled by process 1 which also performs the correlation process on the sampled levels. Process 2 determines the maximum degree of correlation with respect to all permutations of stored correlation functions. This process is shown as representative of a test procedure.

The general nature of the test for correlation is based on searching the stored correlation function to achieve a "best fit" for the particular desired specie. Processes 1, 2, and 5 form the closed loop iterative procedure for generating the composite column specie density value which is operated on in block 3.

Process 3 uses the data gathered as a function of altitude (Process 4) and knowledge of the particular correlation function used (Process 5) to produce a density value normalized to a standard atmospheric mass unit. This calculation is performed whenever a new shell altitude is attained. (See Figure 3.4-4).



$$h = (R+A) \sin \phi - R$$

$h_i$  = MEDIAN ALTITUDE  
DEFINING ITH SHELL

FIGURE 3.4-4 EXAGGERATED SHELL STRUCTURE



In this way a series of concentration is produced to reflect the average specie concentration as a function of altitude. These limb measurements are time constrained to the vehicle transition through its local dawn and can occur only once per orbit due to the physical mounting of the sensor on the vehicle mainframe.

Processes 6, 7, 8 and 9 form an identical procedural loop with the following two exceptions: First a different set of correlation functions are used in this nadir measurement and secondly, only column densities can be computed because of the lack of altitude information. Process 9 produces the normalization factor so that the concentrations appear in normalized atmospheric mass units. As previously mentioned, the non-thermal readings do not require temperature correction.

Blocks 1, 2 and 3 are identical to the non-thermal measurement processing tasks previously described.

Processes 4, 5, and 6 depict the functions required for the strongly temperature dependent thermal nadir measurement requiring an accuracy of  $1^{\circ}\text{K}$  to properly separate the contribution of a specie of gas from the total superimposed interferograms. Based on an empirical temperature profile, a set of mixing ratio of gases is selected. A correlation function is then chosen which generates an estimate of the given gas specie. The estimated value is compared with a model value so that the difference generates a new gas density for the mixing ratio which, with the known temperature profile, allows for the selection of a second correlation function. This iterative procedure is continued until an accurate estimate of the measured gas value is obtained. Although there are several techniques available from estimation theory, the specific algorithm is still in the selection process. For the purpose of this study a minimum distance classifier is assumed.

## HARDWARE/SOFTWARE TRADE OFF

The CIMATS instrument is a low frequency sensor i.e. 116 wds/sec including the processing of the temperature profile from an auxilliary temperature sensor (A radiometer), so that processing is not rate constrained. Current investigatory work is centered around a mini-computer with a 32K x 16 core memory with estimation of 8 to 12K of the 32K to store the correlation function coefficients and 1K allocated to the program.

Based on the flow diagram the major process to be performed is:

$$\langle S_n \rangle = \sum_{k=1}^N A_k I_k$$

This interpolation is relatively easy due to apriori known value of  $A_k$  so that only real time multiplication and accumulation are required.

Processing requirements for the CIMATS Data reduction flow also require trigonometric and inverse trigonometric capability. The use of such functions is extensive as they are needed only on a per orbit basis.

The bulk of the instrument utilization is planned for the nadir measurement mode. Therefore, it is necessary to process the thermal band interferogram data though the loop shown in Figure 3.4-5 (Process 4, 5, 6). The nature of the selected algorithm has a direct impact on the complexity and, hence, practicability of performing the data processing in the vehicle.

Initial estimates for this overall process involve no more than a few hundred operation. At a conservative estimate of 10 usec/step, the total time required places no burden on the OEDSF.

The most demanding requirement for an onboard processor is the memory for the 160K bits for the correlation coefficients. By using low power semiconductor technology, this is accommodated with a power demand of less than 1 watt and implemented on 16 LSI IC's.

In considering the means of implementing the CIMAT Processing flow, the choices are more appropriately taken between firmware and software rather than hardware and software. Only logical comparison and limited iterative arithmetic operations are required. A firmware approach is preferred for reasons discussed below.

The firmware approach requires about 30 to 40 MSI parts plus the required ROM previously described and closely resembles a primitive three register machine. This approach has the advantage that it is dedicated.

The final choice will be determined by the general architecture selected for the OEDSF, and the economics of generating firmware versus software for a General Purpose machine.

ORIGINAL PAGE IS  
OF POOR QUALITY

# CIMATS DATA PROCESSING

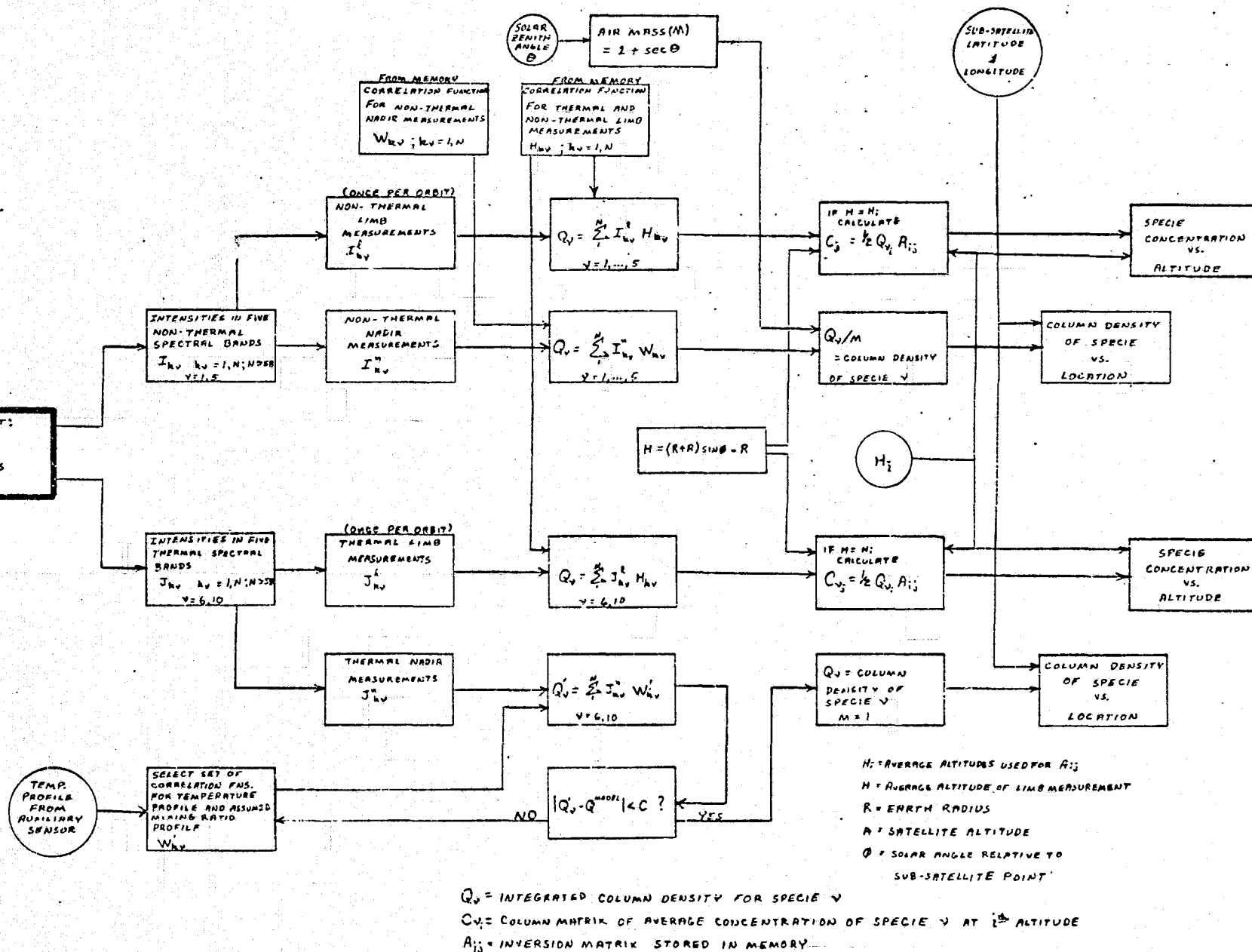


Figure 3.4-5

# CINATS

SUB LAMB MEASUREMENT LAT AND LONG.

ORIGINAL PAGE IS  
OF POOR QUALITY

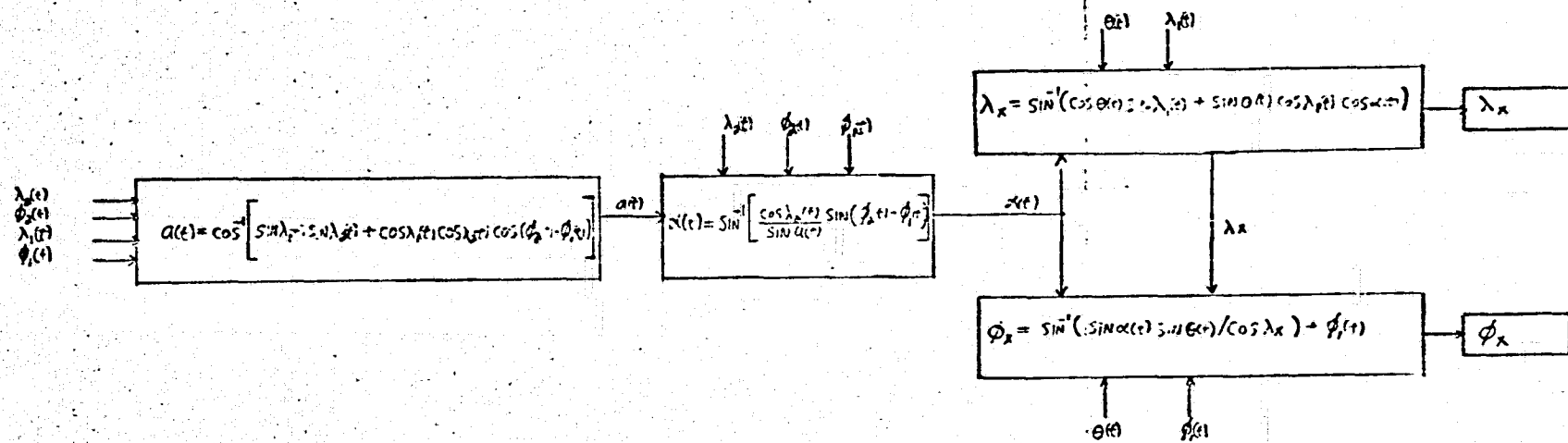


Figure 3.4-6a

ORIGINAL PAGE IS  
OF POOR QUALITY

CIMATS  
NON-THERMAL

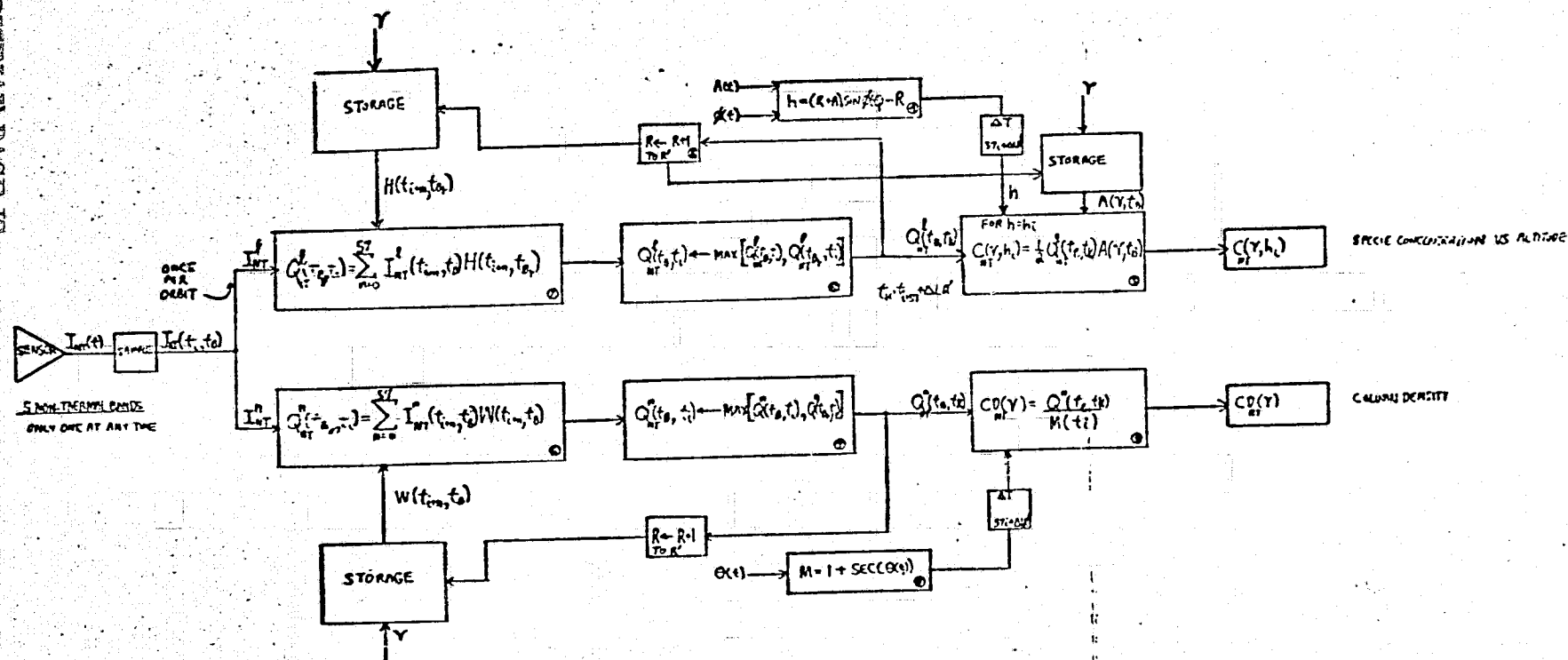


Figure 3A-6b

ORIGINAL PAGE IS  
OF POOR QUALITY

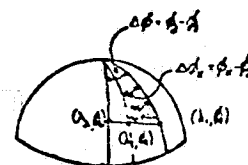
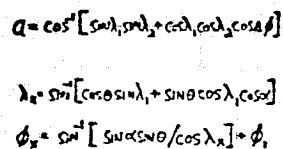


Figure 3.4-6c

$\lambda_1$  - SWR - CATELITE LAT.  
 $\phi_1$  - SWR - CATELITE LONG.  
 $\lambda_2$  - SMOKE LAT.  
 $\phi_2$  - SMOKE LONG.

#### 4.0 OEDSF DEFINITION

The definition of the OEDSF is the set of functions it must perform. This set was determined by a logical sequence of steps described in this section.

The initial step was to partition the flow diagrams described in Section 3 into Onboard and Ground Segments. The partitioning for each sensor and the criteria used in the exercise are described in 4.1. The onboard segment then imposes the requirements of the onboard processing. The processing requirements of the four boundary sensors were summed and decomposed to provide more general functions. Combinations of these simpler functions can then produce many more processes applicable to the requirements of a considerably greater number and type of sensors than the boundary sensors. These are discussed in 4.2. The decomposing of the functions is in Appendix A.

The ground segment is examined in 4.3 to ensure that neither the partitioning nor the selection of the onboard processing techniques create increased burdens which would, from an end-to-end system aspect, negate the advantages of onboard processing.

An evaluation of the OEDSF is performed in 4.4. This evaluation is qualitative pending the cost estimates to be derived in Task 4. It establishes onboard/ground guidelines (which were utilized in performing the partitioning), and examines some aspects of the OEDSF which must be addressed to eliminate their potentially detrimental features.



#### 4.1 ONBOARD/GROUND PARTITION

The determination of which portion of the data processing system is performed onboard and which on the ground for each of the boundary sensors is a pivotal task in the study.

The approach taken to perform this task was to establish a set of criteria, then perform the partitioning based on these. These criteria were derived by an iterative approach which modified, deleted, or added new criteria in accordance with the results of the evaluation of the system resulting from their application.

The final set of criteria developed during this exercise is as follows:

- 1 - Processing performed onboard by the OEDSF should satisfy all the users of the data. OEDSF processing stops where different users begin to process the data differently.

Many experiments gather data which can be used in several ways. In most cases, fundamental calibration and correction processes and the extraction of basic information is common to all uses. Additional processing is peculiar to the specific use. For example, surface temperature information is utilized and processed differently when it is used for meteorology, crop yield estimation, or energy balance studies (Albedo). The OEDSF is an effective device when it performs processes common to all users since it eliminates the duplication of these processes by the individual users, or expedites delivery of their data by avoiding the delay they would incur if these common processes were performed in a single ground facility following the return of the shuttle. Further, the chief benefits derived from onboard processing (real-time availability of ancillary data, for example) tend to be realized in the primitive processes, which usually are also the common processes.

- 2 - All onboard processing will be on-line in real or near-real time. Data will not be stored for long periods of time and processed in batches. This criterion is derived from two basic tenets of the OEDSF cost-effectiveness concept: It must exploit the features available onboard but not on the ground; it must not perform processes which simply convert ground equipment into flight equipment.

The major feature of onboard processing is the real time availability of ancillary data which includes shuttle location and attitude, instrument characteristics such as pointing parameters and operation (housekeeping), and other calibration data such as sun angle, sun radiance, and the information provided by auxilliary sensors. This feature is exploited only when the real time aspects are utilized. Storing this data and performing batch processing duplicates the operation of present ground processing modes. Further, it requires storage facilities which tend to be large and difficult to qualify for space flight.

- 3 - Processes requiring large quantities of pre-stored data (i.e., look-up) will be performed on the ground. The term "large" is a variable depending primarily on the memory requirements. The criterion derives from the obvious deleterious effects of having to provide large memory capacities onboard. It is supported by the fact that in most cases, the processes requiring these pre-stored data tend to be in the more advanced categories rather than the basic processes which the OEDSF is ideally suited to perform.

- 4 - Processes requiring pre-stored data which must be periodically updated will be performed on the ground; however, infrequent uplinks of updated data which enhances onboard processing is allowable. This criterion is primarily based on the premise that processes requiring regularly updated pre-stored data tend to be the more advanced and specialized processes which no longer

benefit from onboard features. It is recognized that there will be many exceptions to this premise so that, although it is a first order guideline, it is subject to re-examination where it eliminates primitive processes. The cost of providing an up-date feature must be weighted against the loss of the benefits of onboard processing.

- 5 - The location of the onboard/ground partitioning must not require any extensive onboard process to be repeated on the ground. There are frequent instances when the data must be reformatted following a series of processes. The data must also be reformatted if it is to undergo recording or transmission following any portion of this series, then again reformatted prior to and following undergoing the remainder of the series. Examples are domain transformation and resampling. In such instances the entire series should be performed onboard or on the ground. If the initial processes in the series strongly benefit from onboard processing, even though the remainder of the series does not then the entire series should be performed onboard.

Trade-offs must be effected weighing the onboard processing advantages and disadvantages of the initial and subsequent processes versus performing the entire set on the ground.

- 6 - Processes performed onboard must be well defined and not subject to frequent and extensive changes. Experimental and user modeling processes will be performed on the ground. The configuration and qualification of flight equipment is expensive. The benefits to be derived from onboard processing will be realized only if costs are kept within reasonable limits. Frequent changes and modifications requiring extensive rework of the OEDSF will rapidly erode the cost advantages inherent in its functions.

User models are devices intended to measure the validity of a set of theories by correlating measured facts against predictions derived from the theories. As such they are subject to changes and modifications as the measured data modifies the theory.

The output of this study is a conceptual design for an onboard processor. Such a processor cannot be designed when the processes it is required to perform are not defined or are subject to frequent changes.

- 7 - The characteristics of the data at the partitioning interface must be such as to enable efficient continuation of the processing or utilization.

The basic benefit to be derived from the OEDSF is an overall cost effective system. Data delivered to the ground in a state, configuration, or format which imposes additional complex or extensive processes to continue its further processing diminishes the system effectiveness. The data output from the OEDSF must be "clean" in the sense that it is compatible and easily interfaces with the next set of processes, and maintains a minimum profile in terms of format, ancillary information needs, and conciseness.

These criteria are more correctly referred to as guidelines since each is subject to exceptions or modifications for any given set of requirements. In certain cases, some of them are contradictory. For example, the use of frequently updated data may eliminate the repeating of extensive processes on the ground. Tradeoffs between these guidelines may therefore be one of the first steps in partitioning candidate systems. Table 4.1-1 indicates the criteria which may conflict with each other.

CRITERIA	1	2	3	4	5	6	7
1	-	N	N	N	N	N	N
2	N	-	N	N	C	N	C
3	N	N	-	N	C	N	C
4	N	N	N	-	C	N	C
5	N	C	C	C	-	N	C
6	N	N	N	N	N	-	C
7	N	C	C	C	C	C	-

N = NO CONFLICT  
C = POSSIBLE TRADE-OFF

COMPATIBILITY OF ONBOARD/GROUND CRITERIA  
TABLE 4.1-1

A criterion which provides guidance as to allowable onboard processors size, power and memory requirements is conspicuous by its absence. It became evident that any assignment of quantitative values to these items would be unnecessarily restrictive on the onboard segment at this time. There are obviously limits for these parameters on the OEDSF as an entity; however these will be a function of the sum of all the processes required by all the serviced sensors and the apportionment of space and cost to the OEDSF which will, to a large extent, be determined by its value. These limitations will create trade-offs between the extent of onboard processings for given sensors and the number of sensors serviced, for example. Thus, in the process to establish the desirable OEDSF capabilities it is reasonable to exclude from onboard consideration only those processes whose physical needs are obviously excessive, such as a gigabit memory.

The onboard/Ground partition effected for each of the boundary sensors is discussed below.

The rationale for each system is indicated below and correlated with the applicable criteria on table 4.1-2

ATS- The onboard processing consists of all pre-processing of the data. This includes Calibration, Radiometric Correction and Geometric Correction. The Geometric Correction encompasses X and Y correction based on GNC data providing information on the shuttle attitude and altitude, and on an Earth Model providing information on earth curvature and rotation skew. Ground Control Point (GCP) Correlation is also performed onboard even though this process does not benefit from any inherent onboard processing advantage. The major reason for this decision is that the data must be resampled prior to recording or transmitting to the ground. If GCP correlation were performed on the ground, an additional resampling process would be required following this correction. A double resampling process introduces radiometric errors which reduce the radiometric accuracy below that desired for many applications. Information Extraction processing is performed on the ground because the optimum approach to this task is dependent on the user; i.e., the process to extract wheat acreage is different from that to highlight geological features.

IRS- The onboard processing consists of all processing required to derive the raw temperature profile and mixing ratio profile as a function of position. The process is carried this far onboard because the position data required in these computations is readily available in real time. The temperature analysis is performed on the ground because this process requires a complete reference of the previous day's temperatures for each subgrid point at each altitude level. The output of this process is a set of plots (one for each altitude). The process gains nothing from being performed onboard and is more efficiently performed with large general purpose computers.

CRITERION

EXPER.

1

2

3

4

5

6

7

ATS	Processing restricted to calibration and preprocessing (calibration, geometric and Radiometric correction)	Modified GCP correlation to predictive approach using Kalman filter	N/A	Ability to perform GCP correlation with 3 points per frame reduces req'ts for update to tolerable quantity	Perform GCP correlation onboard to avoid a second resampling process	No information extraction performed onboard	Achieved by implementation data XMTD is geometrically and radiometrically correct.
IRS	Output of OEDSF is raw temperature profiles	New techniques developed to average cal values before and after data	Temp. analysis requiring previous day's temperatures performed on ground	Modification of process allows onboard processing of function without ground update of previous day's temperature	Onboard process obviates need to convert radiance to temp. and vice versa in subsequent processing	Final temp. analysis performed on ground	Data XMTD is temp. as a function of lat. and lon.
RADSAT	Output of OEDSF are $\epsilon_0$ and $T_a$ which are basic values	extensive utilization of real time ancillary data	N/A	N/A	N/A	Utilization of $\epsilon_0$ and $T_a$ for various models performed on ground	Data XMTD is $\epsilon_0$ and $T_a$ as a function of lat. and lon.
CIMATS	Output of OEDSF is specie concentration	extensive utilization of GNC data in real time	Stored interferograms are deemed small and vital to beneficial onboard processing	N/A	Complete processing onboard to obtain specie concentration eliminates further ground processing	N/A	Data XMTD is specie concentration as a function of lat., lon., and altitude

IMPACT OF ONBOARD/GROUND CRITERIA ON BOUNDARY EXPERIMENTS PROCESSING

TABLE 4.1-2

RADSCAT - The onboard processing consists of the computation of the backscatter cross-section ( $\sigma_0$ ) and the antenna temperature ( $T_a$ ) as a function of position (latitude and longitude). Subsequent processing is performed on the ground for a couple of reasons. First, there are several parameters requiring differing processes which can be derived from these two values; second, the procedures for determining these parameters are presently not well defined.

CIMATS - The entire processing of the Cimats data yielding specie concentration as a function of altitude and location is performed onboard. Any subsequent processing involves user models.



#### 4.2 OEDSF REQUIREMENTS

This section establishes the data processing requirements of the OEDSF.

The requirements are derived from the onboard segment of the functional flow diagrams in Section 3. These boundary sensors, by definition, establish both the spectrum extremes for signal characteristics and the extremes of the processing complexity.

The OEDSF must handle many experiments from several disciplines, thus the processing requirements established by the boundary sensors must be generalized, and the processing capability of the OEDSF derived from these requirements must be implemented with sufficient flexibility to perform more than these processes.

The approach taken to determining OEDSF requirements which satisfy this objective is described below. The closed functions depicted in the functional flow diagrams are not generally the process requirement. These closed functions are the mathematical relationship which the OEDSF must model. Consequently, each relationship must be described as a set of functions interrelated and generally termed an algorithm. Ramifications result based on the level of decomposition of the closed function. The depth of the decomposition is a variable which must be selected to optimize the combination of the conflicting objectives of general purpose and low cost. If the decomposition is too shallow, a special purpose, sensor unique function, results. If the decomposition is too deep, a general purpose machine results that is too cumbersome from an implementation and user standpoint.

The detailed work performed in this task is contained in Appendix A.

The required processing functions tabulated on the flow diagrams were extracted and converted to an implementation process; i.e., the actual process which will

implement the required function. Algorithms were then developed to perform this process. The steps of the algorithms were then grouped as the set of functions required.

Requirements which create only functions already developed are not considered again. The vast majority of functions developed in Appendix A were provided by the early processes of the CIMATS and the IRS. The only new functions supplied by the Radscat, for example, was Matrix Multiplication. Processes required in handling housekeeping and command data were also considered and found to be well within the envelope defined by the data processes.

The required functions were generalized and grouped into process categories.

Table 4.2-1 tabulates the 18 functions derived from Appendix A grouped into the four process categories.

Table 4.2-2 relates function groups to the sensors whose processing requirements use one or more of the functions in the group.

## OEDSF FUNCTIONS REQUIRED

### 1. TRIGONOMETRIC FUNCTIONS

- |              |                    |
|--------------|--------------------|
| a. Sine      | f. Cosecant        |
| b. Cosine    | g. Inverse Sine    |
| c. Tangent   | h. Inverse Cosine  |
| d. Cotangent | i. Inverse Tangent |
| e. Secant    |                    |

### 2. EXPONENTIAL FUNCTIONS

- a. Exponential
- b. Natural Logarithm

### 3. ALGEBRAIC FUNCTIONS

- a. Algebraic addition with accumulation capability
- b. Signed multiplication with reciprocal input capability

### 4. CONTROL FUNCTIONS

- a. Multiplexing
- b. Demultiplexing
- c. Storage and Retrieval
- d. Counting
- e. Delay

TABLE 4.2-1

SENSORS PROCESS REQUIREMENTS

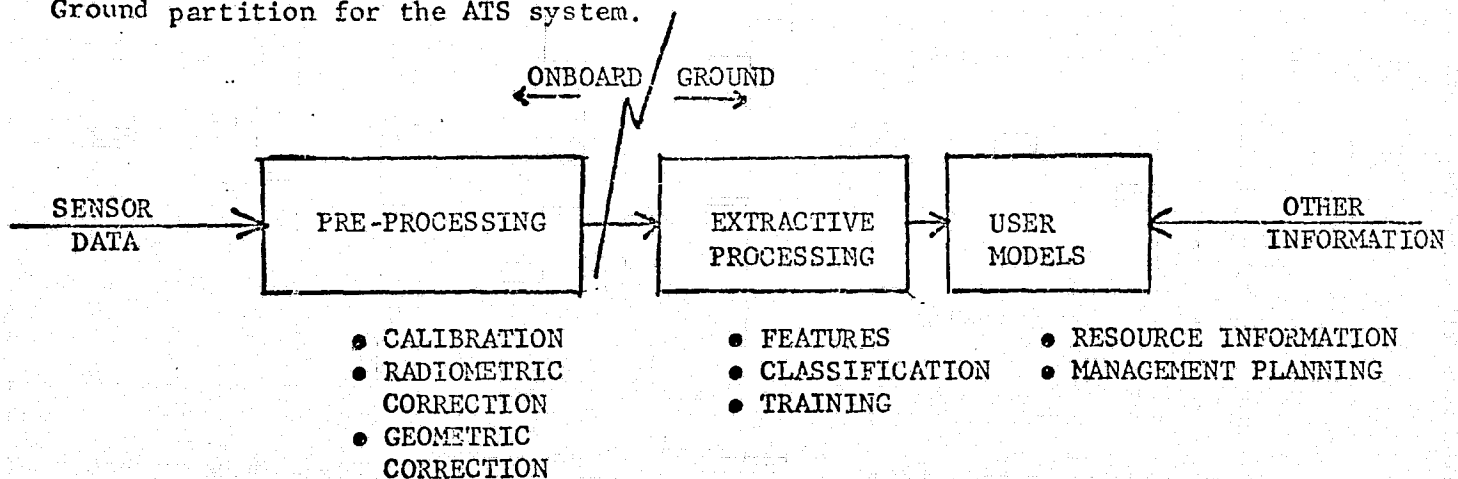
<u>PROCESS</u>	<u>SENSOR NEEDS</u>			
	<u>CIMATS</u>	<u>RADSCAT</u>	<u>IRS</u>	<u>ATS</u>
TRIGNOMETRIC	YES	YES	NO	YES
ALGEBRAIC ADDITION WITH ACCUMULATION CAPABILITY	YES	YES	YES	YES
SIGNED MULTIPLY WITH RECIPROCAL INPUT CAPABILITY	YES	YES	YES	YES
EXPONENTIALS	NO	NO	YES	YES
CONTROL	YES	YES	YES	YES

TABLE 4.2-2

### 4.3 OEDSF GROUND SYSTEM REQUIREMENTS

This section examines the requirements imposed on the ground segment of the four boundary experiments data systems as a result of the partitioning. The intent of this examination is to enable a gross evaluation of the effectiveness of the entire system to ensure that processes performed onboard and the location of the onboard/Ground partition do not reduce the advantages of onboard processing by creating new and extensive processing requirements on the ground.

ATS - The data provides information useful to many disciplines such as agriculture, forestry, geology, urban planning, and hydrology. The information required is extracted from data provided in several spectra, over a period of time, and correlated with other information obtained from exogeneous sources. Figure 4.3-1 depicts a generic data processing system indicating the Onboard/Ground partition for the ATS system.



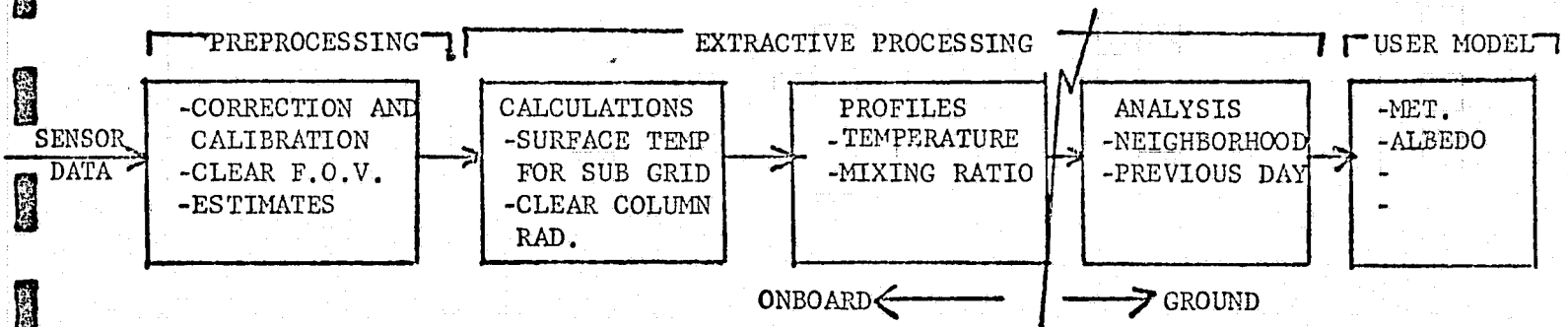
ATS DATA SYSTEM  
FIGURE 4.3-1

The ATS data input into this system undergoes several processes which render it useful for the particular application. These are, in general, a function of the specific application; however, all applications share a common need which define the basic processing of the data. These are: calibration, radiometric correction, and geometric correction. These processes will be performed onboard; all subsequent processes will be performed on the ground. The data supplied to the ground is radiometrically and geometrically corrected digital data. The processes which may then be performed on the ground are as various as the uses of the data.

Typically they consist of information extraction which may be performed by thematic techniques, typified by the Image 100, an interactive thematic extraction processor. (The reader is referenced to the OEDSF Task I report, Pages A-41 to A-49). This is followed by user modeling which combines this information with information obtained from other sources to create a final output product. For example, ATS data providing information on crop acreage and health may be combined with meteorological information providing temperature and soil moisture to determine crop yield.

The specific ground requirements which may be specified relate to the input interface. The output of the OEDSF will be a High Density Digital Tape (HDDT). The ground facility must be capable of converting this tape to a Computer Compatible Tape (CCT) or directly to imagery. These requirements would exist without the OEDSF, since raw ATS data would be recorded on an HDDT. The ground segment requirements of the system are thus reduced to the extractive and user model requirements by the elimination of the need to preprocess the data.

IRS The IRS provides atmospheric temperature profiles and earth surface temperature as a function of location. This information may then be used in user models to support various disciplines, in particular, meteorology. The basic process to extract the information from the sensor data, and the location of the onboard/Ground partition are indicated in figure 4.3-2.



IRS DATA SYSTEM  
FIGURE 4.3-2

The data delivered to the ground are the raw temperature profiles and mixing ratio profiles as a function of location.

The ground system must perform the surface temperature analysis. This process is identical to that performed at present on the HIRS data, and the same program developed for that phase of the processing may be used. One step has been added as a result of the method used to implement the onboard processing. As indicated on the IRS flow diagram, section 3.2, if an insufficiently clear field of view exists in a sub-grid, a flag is set, and a bilateral estimate temperature value based on the average of the four nearest qualifying neighbors is used in further processes. The present approach (all ground) is to use the previous day's temperature for this sub-grid. The use of the estimated temperature instead of the previous day's temperature in the data processing produces at worst a second order error; the estimated temperature can be replaced with the previous day's temperature during the

analysis operation by a simple modification of the existing program. Presently the analysis program performs various checks on each sub-grid temperature and replaces it with the previous day's temperature if it fails any of these. The modification consists solely of adding a flag set check to the other checks, and considering a set flag as a check failure.

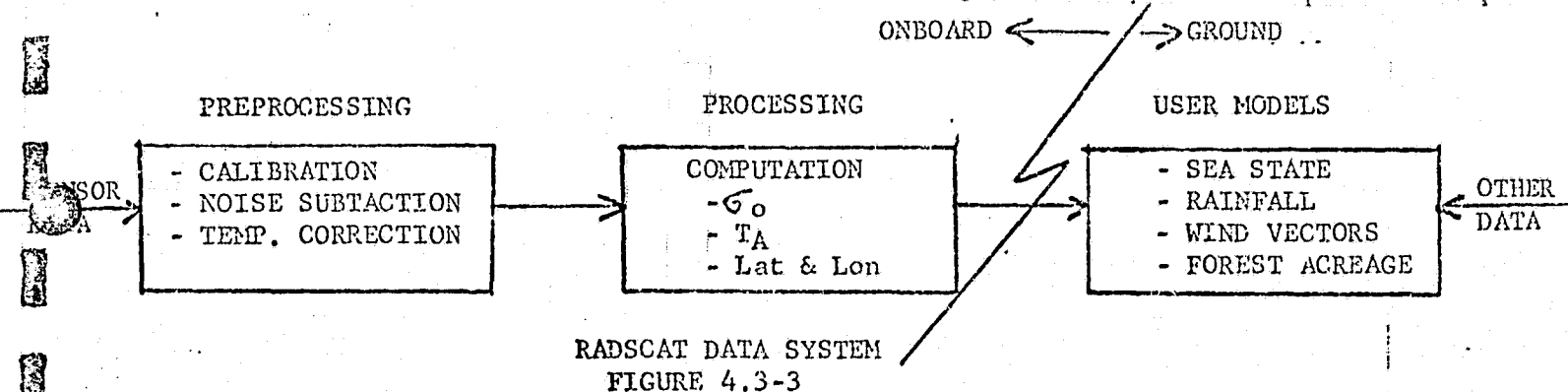
All other ground processing, including user models are unaffected and retain their present requirements.

### RADSCAT

The RADSCAT is an instrument consisting of a radiometer and a scatterometer which produce data from which basic parameters of the target may be derived. These basic parameters are, the backscatter crosssection ( $\sigma_0$ ), and the target temperature ( $T_t$ ). The computation of the target temperature is based on the Radiometer antenna temperature ( $T_a$ ) and uses several other data (which may include  $\sigma_0$ ) obtained from exogeneous sources. The complexity of this process, depends on the accuracy of  $T_t$  desired. For several applications  $T_a$  is sufficient; thus the computation of  $T_t$  is a user model function. These two parameters may be used singly or in conjunction with each other (or with other data) to produce information on several characteristics of the target. Examples of information derived from these parameters are: Sea wave height, wind velocity and direction, soil moisture, crop stress, geological surface features, water salinity and temperature, and forestry management parameters.

The generic data processing diagram for the RADSCAT is shown in figure 4.3-3.





The present RADSCAT ground system consists of two basic entities. The preprocessing and processing are performed at a central facility. The output of this facility are  $G_o$  and  $T_a$  as a function of position. This data is distributed to various users most of whom are presently in the experimental phase; i.e., developing and evaluating models which produce the final information.

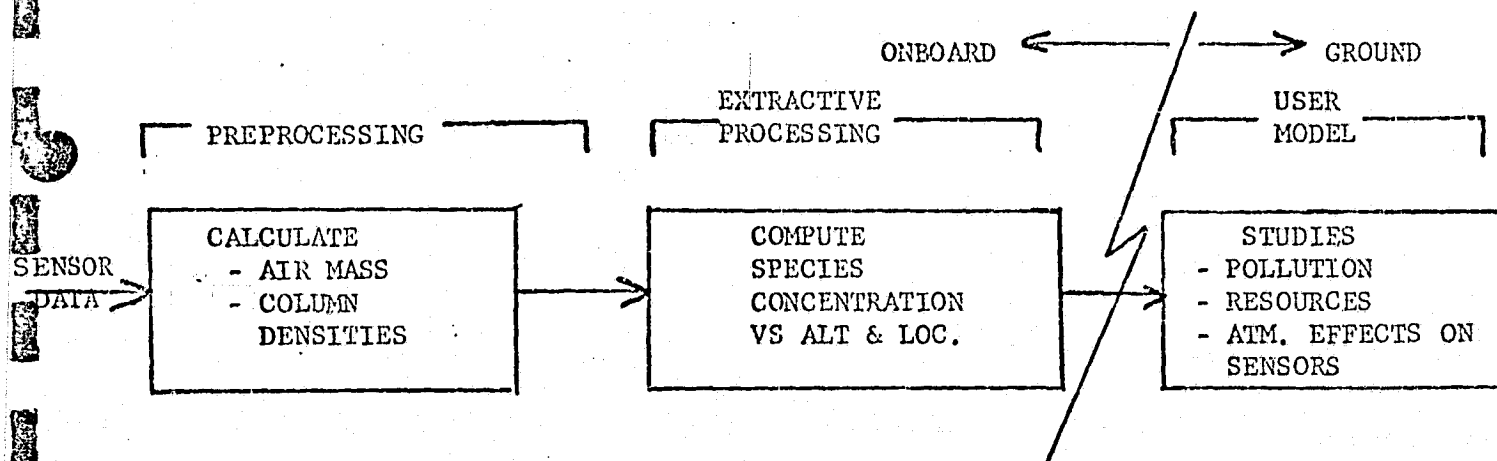
The OEDSF performs the preprocessing and processing functions and outputs the identical product as that supplied by the present facility; hence, there is no impact on the user models and subsequent processing of the RADSCAT data by the OEDSF.

#### CIMATS

The CIMATS produces data which enables the determination of the column density of a number (approximately 9) of gas constituents of the atmosphere as a function of altitude and location.

The initial utilization of this information is the study of pollution. Corroborative measurements made from the ground are used in this study. There will undoubtedly be many other uses of the CIMATS information related to the concentration of various gases singly or in group. These are all user model functions.

The generic processing flow of the CIMATS data is shown in Figure 4.3-4.



CIMATS DATA SYSTEM  
FIGURE 4.3-4

The CIMATS pre-processing function is unique in that it is really the early phases of information extraction rather than the more classical calibration and correction functions associated with this term.

The entire information extraction process is performed onboard. The input to the ground system is the specie concentrations as a function of altitude and location. These are submitted to the user models which are undefined at this time. The format of the supplied data will be High Density Digital Tapes.

#### 4.4 SYSTEM EFFECTIVENESS EVALUATION

This section evaluates the consequences of performing the selected processes on board. The intention of this evaluation is to determine, on a gross level, if the processes performed on board improve the overall system with respect to cost effectiveness and timeliness of data availability to the experimenter.

A more detailed evaluation which includes quantitative cost estimates for the OEDSF and ground segments will be performed in Task 4.

The OEDSF realizes its benefits by exploiting its unique location in both a spatial and temporal sense. This exploitation is enhanced by the judicious choice of the processes which it performs, and by its architecture.

Table 4.4-1 summarizes factors considered in the evaluation (note that the "TBD's" will be determined in Tasks 3 and 4). For each of the boundary sensors, the OEDSF produces data or information ready for extractive processing or user modeling. In each case, the processing requirements on the ground are significantly reduced or eliminated. These advantages are traded off against the cost of developing and programming the OEDSF, and the possible inconvenience associated with developing instruments while having limited access to a portion of their data processing equipment.

##### Temporal Advantages:

The OEDSF operates in real time. The output signals from the experiments are fed to the OEDSF as the experiments generate them. All ancillary data is available to the OEDSF coincident with its generation. Ancillary data is

	DATA IMMEDIATELY AVAILABLE ON (HDDT)	DATA COMPRESSION RATIO	ANCILLARY DATA	GROUND PROCESSING ELIMINATED	GROUND PROCESSING ADDED	PRESENT APPROACH		PROPORTIONAL COST OF OEDSF
						TIME	COST	
ATS	CORRECTED DIGITAL IMAGERY WITH LAT & LON	NONE	ELIMINATED	CALIBRATION RADIOMETRIC & GEOMETRIC CORRECTION	NONE	20 MIN/SCENE (100 x 100 NM)	\$3M/STA	TBD
IRS	RAW TEMPERATURE AND MIXING RATIO PROFILES WITH LAT & LON PER GRID	16:1	ELIMINATED	CALIBRATION CALCULATION OF TEMP & MIXING RATIO	FLAG CHECK	TBD	TBD	TBD
RADSCAT	$\sigma_0$ AND $T_a$ WITH LAT & LON	90:1	ELIMINATED	CALIBRATION CALCULATION OF $\sigma_0$ AND $T_a$	NONE	TBD	TBD	TBD
CIMATS	SPECIE CONCENTRATION WITH LAT, LON, AND ALTITUDE	20:1	ELIMINATED	ALL	NONE	TBD	TBD	TBD

OEDSF EVALUATION FACTORS

TABLE 4.4-1

ORIGINAL PAGE IS  
OF POOR QUALITY

all data used to operate upon or characterize the experiment data. It includes the following:

- Housekeeping data which provides information on mode, status, and environment. As an example, the RADSCAT processing equations include the antenna housing temperature.
- Guidance, Navigation, and Control (GNC) data which provides information on all Shuttle locations, attitude, and the rate of change thereof - this data produces the location of observed phenomena, which is a requirement of all experiments.
- Auxiliary information. This is information which may be produced by other sensors, for example, the IRS data can be used to correct ATS data for atmospheric effects; or it may be the utilization of ambient characteristics; for example, calibrating the ATS by measuring the sun disk.

If this ancillary data is not utilized in real time, it must be recorded for subsequent processing. The recording process requires a formatting and a time-tag operation of both the sensor data and ancillary data; the subsequent processing requires a correlation operation to "re-match" the ancillary data with the sensor data. Alternately, the ancillary data may be multiplexed with the sensor data so that re-correlation is obviated, but a more complex formatting and reformatting process is required; further, each sensor must duplicate the recording of this common information with a corresponding multiplicative effect on the recording burden.

The real-time feature of the OEDSF provides an adaptive property to the collecting and recording of data. Some examples of the utilization of this property are:

- Inhibit recording of bad data (such as cloud covered targets, or when SNR is inadequate).
- Select signals to be processed (or recorded) from multi-signal or multi-channel instruments based on criteria which may be dependent on the scene characteristics or the signals characteristics.
- Establish or change instrument operating mode based on characteristics of data or ambient.
- Vary the rate of correction data collection based on the measured rate of change of the error inducing agent.
- Point instruments.

Processing the data prior to recording or transmission usually effects significant reductions in recorded volume. The ancillary data which need no longer be recorded often exceeds the volume of data produced by the low frequency (up to several kilobits per second) sensors.

As the prime data gets converted to information, its bulk greatly diminishes. For example, the IRS raw data is collected in 12 bit words for each grid point in 17 channels, a total of 17, 136 bits for each group of 3 subgrids (28 points per subgrid). The output of the OEDSF is 20 temperature values and 20 mixing ratio values at 7 bits each for each group of 3 subgrids, for a total of 280 bits, a compression ratio greater than 16 to 1.

The most significant aspect of real-time processing is that the data is ready for the experimenter when the shuttle lands. The pre-processing through a central facility with its attendant queue is eliminated.

#### Spatial Advantages:

The OEDSF derives advantages by virtue of its co-location, in space, with the instruments. One obvious benefit is that processes common to all instruments (cloud cover, spacecraft position and attitude, atmospheric conditions, etc...) need be performed only once. If they were performed on the ground, they would be repeated at each experimenter's site, or they would be performed at a central facility with its attendant queue (up to one year on Skylab

The major benefit lies in the sharing by the instruments of the OEDSF's set of processing functions. The judicious decomposition of the processes required by the various instruments yields a finite and limited set of basic functions which, in various combinations, satisfy the processing requirements of all the sensors. The level of processing capability of

each member of this set is sufficiently high that the programming associated with their combination is simple (and inexpensive). The development of such a set for a single experiment would be prohibitively expensive. It becomes highly cost effective, however, when several experiments simultaneously share the same functions. The architecture of the OEDSF (see Section 5) has been configured to maximize the benefits derived from these circumstances.

#### Cost of the OEDSF

This subject can be treated only in a qualitative manner pending the cost estimates which will be performed upon completion of the conceptual design. The OEDSF will be an advanced, sophisticated, flight qualified processor. It will be cost effective by virtue of the savings accumulated from the many ground systems it replaces, and the immediacy of its output's availability to the user.

The OEDSF's architecture permits a modular growth. The full payload version can be evolved as the benefits of onboard processing are certified on limited versions.

The OEDSF will incur recurring costs each time the complement of sensors it services is changed. These costs are expended in the reprogramming of the array. As discussed in Section 5, the nature of the architecture coupled with a carefully tailored language will keep this cost low: orders of magnitude below that required to program a general purpose machine. A skilled specialist will perform this programming. This is both an advantage and a disadvantage: the experimenter must supply a complete and accurate description of the processes required by his instrument. The disadvantage is

that the experimenter loses some freedom in evolving these processes as he experiments. The advantage is that it compels him to think through these requirements before he undertakes the expensive spaceflight phase of his experiment's development.

#### Instrument/OEDSF Integration

The OEDSF is a central facility and is not available on a full time basis to each experimenter at his site. This causes two distinct challenges: development of the instrument and checking out the instrument for flight without a part of its data processing.

The development of the instrument does not, in general, require the functions performed by the OEDSF. The proper operation of the instrument is normally determined by analysis of its raw output. The calibration, correction, and data reduction functions are system level operations which may be simulated as necessary. Typically, the ancillary data required for these processes are simulated in primitive forms when these checkout phases are performed on the ground. Hence, the unavailability of the OEDSF during these operations is not a significant factor.

The integration of the instruments with the OEDSF is a major activity. Many instruments will converge on the Shuttle (and the OEDSF) within a short time period. Each and all of these instruments must mate and operate with the OEDSF (and with each other). The successful realization of this endeavor is the greatest challenge to the OEDSF concept. It will happen only if thorough and detailed plans are formulated and implemented. These will, again, impose an added burden on the experimenters and must be placed on the negative side of the OEDSF benefits equation. The solution



must not require a duplication of the OEDSF's functions at the user's site. This approach would require significant additional expenses which erode the cost advantages of the OEDSF. Recommendations which address this challenge will be developed during Task 4.

#### Secondary Impacts of the OEDSF

The advent of onboard processing and the method of its implementation create a new environment which affects some facets of experiment development.

Some examples are:

- Better disciplined experimenters. As discussed earlier, the effective utilization of the OEDSF requires that the experimenter fully develop his processing requirements prior to the flight. This forces his attention onto matters which are usually considered secondary creating an attitude which often results in one of two situations: the experimenter omits from his requirements critical ancillary data and thereby renders his experiment worthless, or he requests all the ancillary data he can think of to insure that he will have available whatever he may subsequently need, thereby creating an unwarranted demand on the system.
- The OEDSF requires an OEDSF programming specialist. Many experimenters' data reduction facilities are programmed by either the experimenter or his assistants. They are experiment oriented rather than processor oriented. The requirements for a specialist insures that the programming will be effected in the most efficient and economical procedure possible.
- The OEDSF is flight equipment. In all space systems built to date the ground equipment complement has been treated as a poor second to flight equipment in the areas of planning, management, and allocation of resources. This pattern will not change in the foreseeable future. Data processing has suffered from the fact that it has been a ground process. Data processing, when performed onboard, will benefit from the very significant advantages accorded flight equipment.

The onboard/ground criteria and the partitioning decisions stated in Section 4.1 are conclusions derived from the considerations discussed in this section. They are optimal, given that onboard processing is beneficial in terms of cost effectiveness. This tenet remains to be proven by "hard" numbers, a pivotal result of Task 4.

## 5.0 OEDSF ARCHITECTURE

By definition, architecture is the art or science that pertains to the method or style in which some physical structure is built. In electronic signal processing, an architecture is more explicitly defined as the method of establishing the inter-signal relationship with respect to the processes or transfer functions comprising the system. At the system level, architecture defines the processing philosophy and dimensional distribution. Processing structures are further characterized as functions of time. In describing an architecture, the following terminology will be used.

- o Algorithm - A well defined procedure specified as a sequence of steps that are required to perform a desired function.
- o Event - The occurrence of a Phenomenon, i.e. signal, dependent on both machine state and mode.
- o State - A configuration of the machine at a given time established by the status of its elements.
- o Mode - A sequence of operations determined by the outcome of an event.
- o Iterative - Any process which has periodicity and is normally applied to a block of hardware, software, or firmware.

The initial step in selecting a candidate architecture for the OEDSF at both the processing and system levels, is to rapidly reduce the broad spectrum of available architectures to a reasonable number by eliminating all but the applicable structures.

The general system requirements may be established from the functional flow diagrams in Section 3. Based on these requirements, applicable architectures can be nominated and trade-offs performed to select the best one.

Some fundamental guidelines are listed below:

- o The architecture must be capable of real time processing, in accordance with the Onboard/Ground trade-off criteria.
- o The architecture must be capable of processing multiple inputs because of the multi-sensor nature of the STS missions.
- o The OEDSF must be capable of operating on large data arrays because of the high data rate output by some sensors and the large number of sensors.
- o Since the OEDSF will be an information processor, the system must be capable of performing complex mathematical computations.

It should be noted that the resultant structure of an actual system is not rigidly constrained to the classical descriptor but is usually a composite. There are many levels of architecture present in large systems at the system, sub-system, component, and even the module level; consequently, the architecture classification is that which is the most dominant with respect to the processing of the information. A key concept is that architectures are not classified by the physical components but by the nature of the processing. For example, a system may contain a small computer and a special purpose pipeline. The nomenclature of the architecture is determined by which is the master and which is the slave; i.e., an augmented computer architecture, or a pipeline architecture.

Based on the preliminary requirements, the processing level applicable architectures were reduced to the following:

- o Augmented small computer
- o Pipeline
- o Serial
- o Array

And for the System levels, the applicable architectures are

- o Centralized
- o Distributed
- o Structured

## 5.1 PROCESSING LEVEL ARCHITECTURES

The processing architectures are discussed and compared in the following paragraphs. The candidate and alternate architectures are then selected in conjunction with the system level architectures.

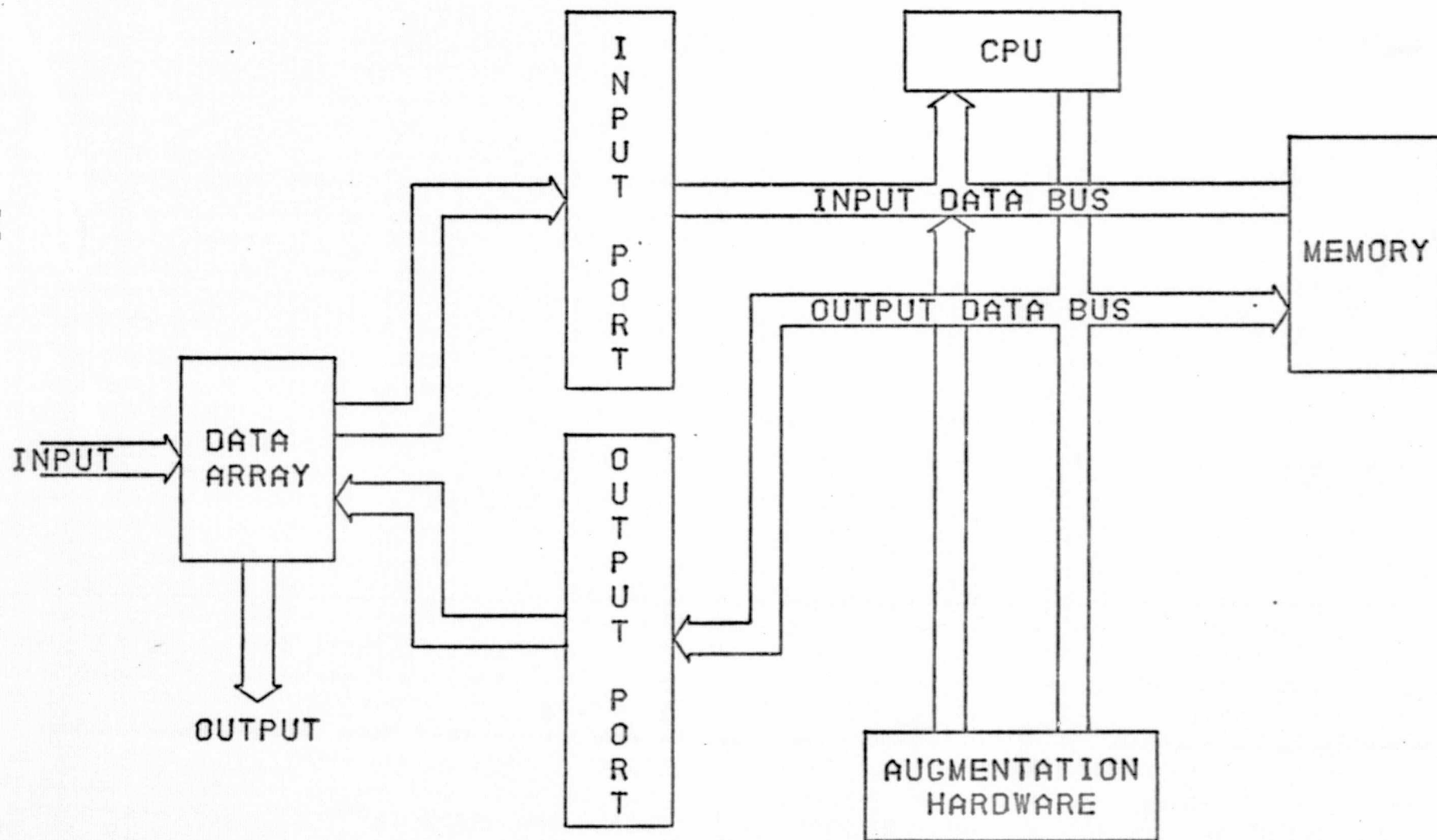
Table 5.1-5 summarizes the characteristics of these architectures

### AUGMENTED SMALL COMPUTER

The augmented small computer architecture is shown in Figure 5.1-1 and its characteristics are summarized in table 5.1-1. The significant feature of this architecture is that any process regardless of the complexity can be implemented in the absence of certain constraints such as time and memory size. In operation, the processor is rather simple due to the presence of the general purpose computer. Data stored in the array is entered into the system by one of two methods i.e. programmed input/output (PI/O) or Direct Memory Access (DMA). Once data has been entered, it is processed in total prior to transmission to the receiving unit. Consequently, the only points of entry and exit for the system are the organic input/output ports of the computer. Since the methods of data transfer limit the speed, this architecture has limited application in real time information processing. Because of the data transfer technique ramifications, PI/O and DMA operation are briefly discussed. Under a programmed I/O transfer, the data in the array is transferred to the computer under control of the computer. This requires a software program and the transfer is accomplished in the following manner: The data in the array is transferred to the accumulator via an I/O port. Once the data is in the accumulator, it may be operated on directly or transferred to another system location. The key is that only one word is transferred from the array to the system at a time.

FIGURE 5.1.1

AUGMENTED COMPUTER ARCHITECTURE



AUGMENTED SMALL COMPUTER

ADVANTAGES

1. ANY ALGORITHM CAN BE IMPLEMENTED REGARDLESS OF THE COMPLEXITY
2. SYSTEM MODIFICATIONS ARE FACILITATED AND DYNAMIC IN NATURE
3. SYSTEM MODIFICATIONS ARE REVERSIBLE AND NOT TIME CONSUMING
4. SYSTEM STRUCTURES, FLOWS, AND INTERACTIONS ARE NOT RIGIDLY DEFINED
5. UNCERTAINTIES MAY BE INCORPORATED, MODELED, AND ALTERED WITHOUT RAMIFICATIONS ON THE SYSTEM
6. DOCUMENTATION IS USER ORIENTED RATHER THAN DESIGNER ORIENTED
7. INTERFACING IS STANDARDIZED AND DOCUMENTED
8. POWERFUL DECISION MAKING AND SEQUENCING CAPABILITY

DISADVANTAGES

1. OPERATIONAL SPEED IS LIMITED BY BASIC MACHINE TIME
2. APPLICABILITY IS DETERMINED BY THE DATA RATE AND FORMAT IN CONJUNCTION WITH THE REQUIRED ALGORITHMS
3. INTERNAL PROCESSING IS SERIAL
4. SOFTWARE IS DEDICATED TO A SPECIFIC SYSTEM
5. ALGORITHM COMPLEXITY ESTABLISHES MEMORY AND POWER REQUIREMENTS
6. MACHINE POWER IS DETERMINED BY THE MACHINE ARCHITECTURE AND THE INSTRUCTION SET

TABLE 5.1-1

Consequently, the PI/O is extremely time consuming regardless of the machine employed. The second method of data transfer is direct memory access or DMA. This accomplishes the transfer of a block of data at the higher memory cycle rate of speed and is controlled by special purpose hardware organic to the computer. The DMA is accomplished in the following manner: The hardware module is programmed with the number of words or bytes (the block length) to be transferred from the data array to the computer memory. Upon establishing the block length, the DMA control hardware is given control of the address and data buses thereby locking the processor out of the system for the duration of the transfer. The data in the array is then transferred directly to the computer memory at the speed of the slowest memory i.e. the speed of the data array or the computer memories. The key aspect is that a block of information is transferred completely bypassing the accumulator, or in reality the CPU, so that no processing can be performed. Some typical and projected transfer times based on a 1975 0.5 microsecond state time are PI/O equal 13.5 microseconds and DMA equal 400 nanosecond for magnetic memory and 400 nanoseconds for semi conductor memory. Projected transfer times (1980) are 1.35 microseconds for PI/O and 40 nanoseconds for semi conductor memory. Computers are state and mode machines where the state time is constant and the mode or cycle time is comprised of a set number of states (typical 5 or 6). Since the machine characteristics are part of the mode and state and designed as such, there is no capability to modify or increase timing without developing a new machine. Consequently, this type of structure cannot process in real time.

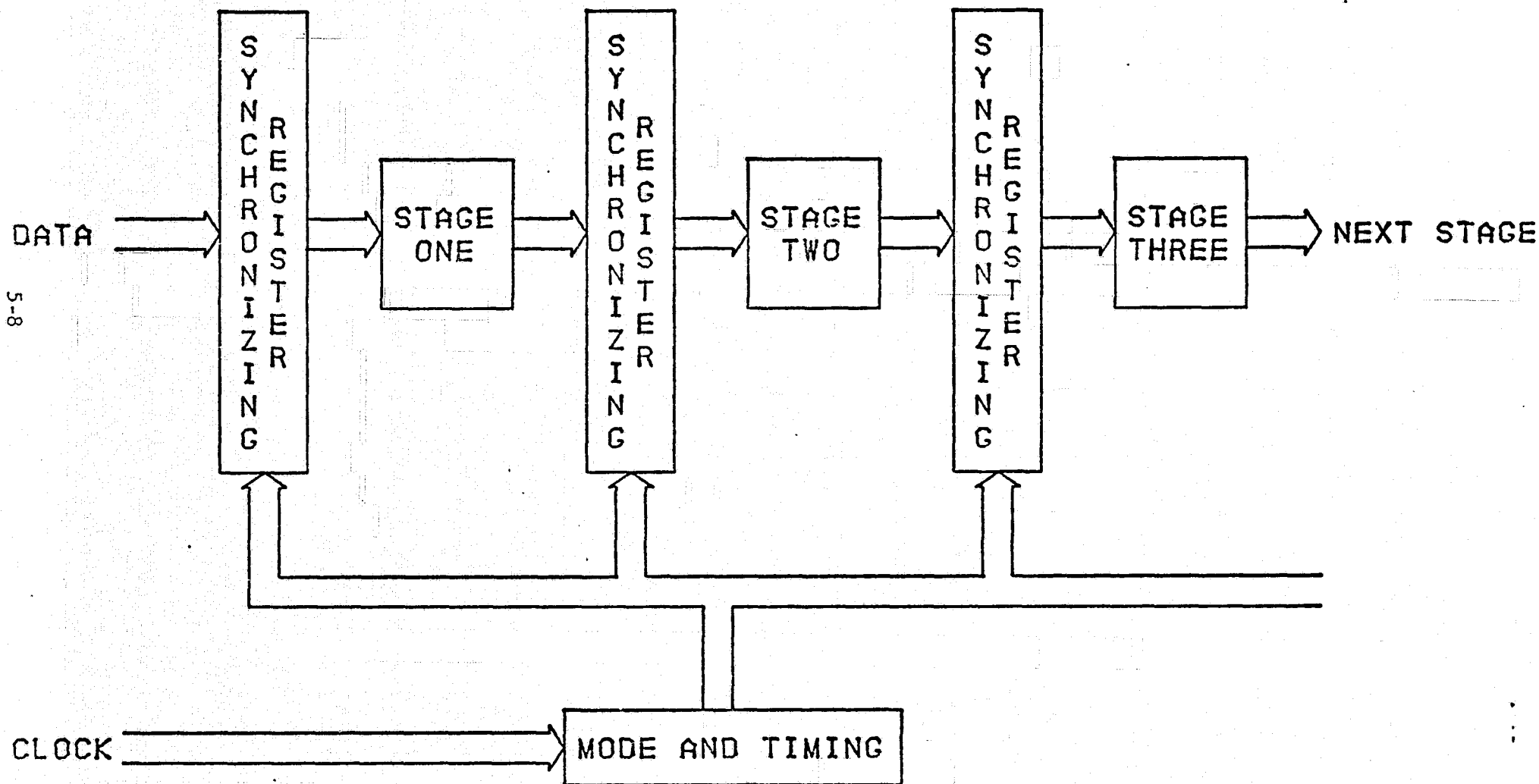
#### PIPELINE

The pipeline architecture shown in Figure 5.1-2 is a high speed, highly efficient structure based on the effective decomposition of the algorithm to the register level for software and to the function integrated circuit for hardware. The general characteristics of this architecture are listed in Table 5.1-2. Although a general machine class, the pipeline is normally employed in special purpose applications. This structure is normally complex and is difficult to comprehend without a thorough



FIGURE 5.1.2

PIPELINE ARCHITECTURE



## PIPELINE ARCHITECTURE

### ADVANTAGES

1. HIGH SPEED PROCESSING DIRECTLY PROPORTIONAL TO THE NUMBER OF STAGES
2. SPEED OF OPERATION IS INDEPENDENT OF THE PROCESSES USED
3. CONTROL OF THE PIPELINE IS SIMPLE AND INDEPENDENT OF THE COMPLEXITY OF THE PROCESSING
4. THE ARCHITECTURE IS MODULAR AT EVERY PROCESSING LEVEL
5. ADAPTIVE TO MATHEMATICAL AND INFORMATION PROCESSING
6. POSSESSES UNLIMITED GROWTH POTENTIAL

### DISADVANTAGES

1. REQUIRES EFFICIENT ALGORITHMS EASILY DECOMPOSED TO SIMPLE SEQUENCES
2. NORMALLY COMPLEX IN DESIGN AND REALIZED IN SPECIAL PURPOSE HARDWARE, FIRMWARE, AND SOFTWARE
3. INEFFICIENT ON SMALL ARRAYS OF DATA
4. THE STRUCTURE MUST BE EITHER OUTPUT COUPLED OR INPUT COUPLED

TABLE 5.1-2

knowledge of computational and signal processing. The structure may be described in the following manner at the high level.

The structure is modeled into  $N$  stages based on the decomposition of the algorithms and the desired rate of entry or exit for the data whichever is higher, and such that the time to perform the required function of each stage is less than or equal to ninety percent of the period of the desired processing rate. Each stage is separated by a synchronizing register which may be any sequential device but is typically a shift register or flip flop type device. Each synchronizing register is driven by the basic clock. Data is then sequenced through each stage where that specific function is performed on the data. For example, at time  $t_0$ , word  $W_0$  is entered into stage  $S_0$  and process  $P_0$  is performed on it, at time  $t_1$ , word  $W_0$  is in stage  $S_1$  so that in general the location of any word in the pipeline is  $M-N$  where  $M$  is the clock count corresponding to the word and  $N$  the scaled pipeline length. The speed is determined by the stage which requires the most time to execute its micro-instruction. This time can be reduced by breaking this function into smaller functions until the algorithm can no longer be decomposed into two or more sequential steps.

The actual hardware used as well as the processes to be performed and their decomposition, will determine the operational frequency. At the present time high speed image processors have been built capable of handling 25 MHz to 125 MHz information word rates.

This structure is speed oriented so that implementation strives to minimize the register operations or delays encountered in processing any data word within one stage. Significant effort, therefore, is normally expended in component selection, modeling, and feasibility evaluations.

## SERIAL

The serial architecture shown in Figure 5.1-3 is an economical structure with the general characteristics listed in Table 5.1-3. This architecture is oriented to operating on large arrays of data which require certain iterative processes.

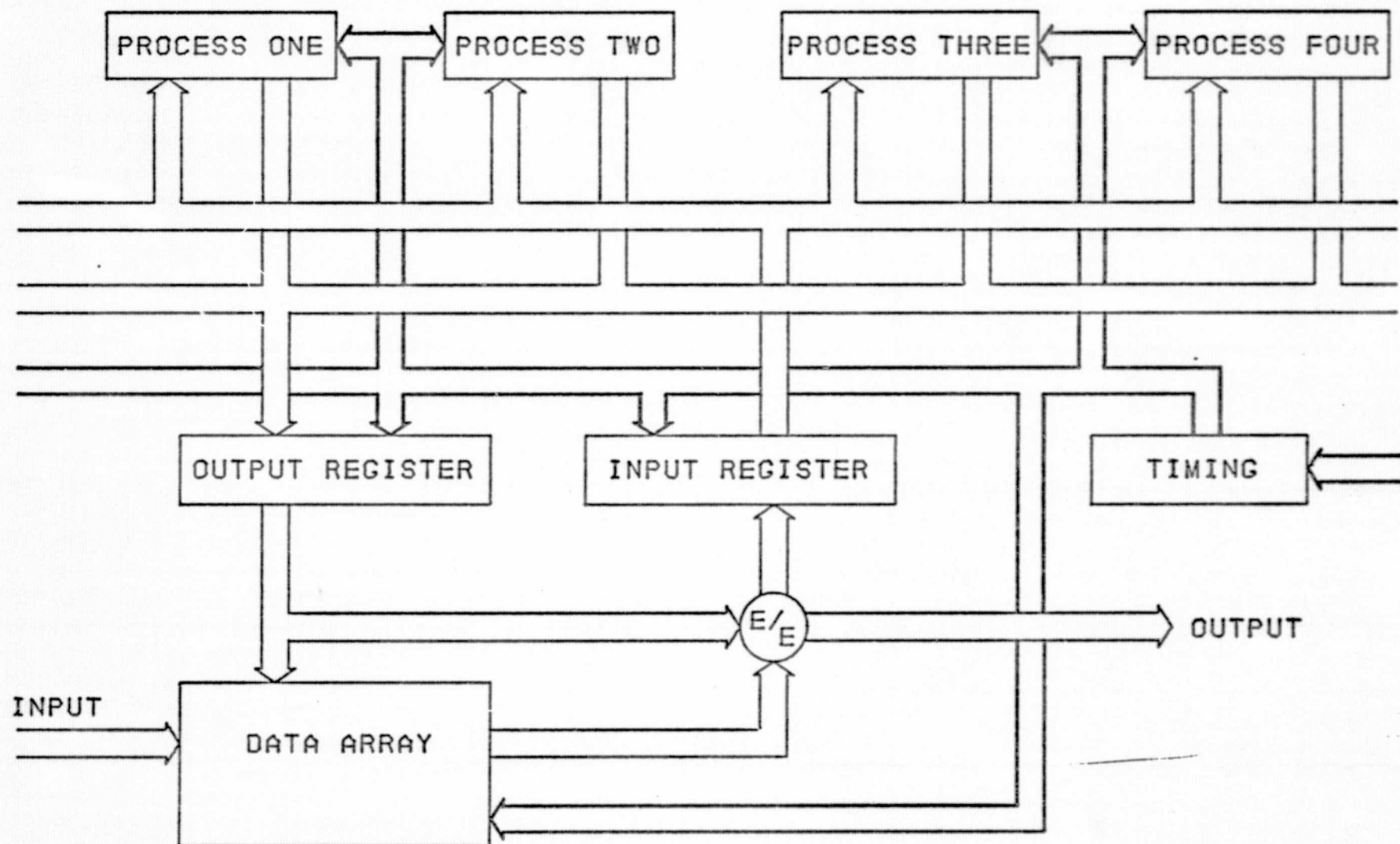
The order in which the processes are to be performed may be randomly ordered so that the structure is in reality a special purpose computer with the processing designed at the register level. The operation of the serial architecture may be viewed in the following manner.

A recirculation loop and loop mask are the key elements of the serial architecture. The recirculation loop sequences through the processing functions while the recirculation mask determines if the function is to be selected. It should be obvious that the entire loop must be traversed if the process desired lies to the left of the polling bit in the recirculation loop.

The data is processed in the following manner: A word is selected by the memory address counter based on the state and mode of the machine and routed to a specific process. Upon completion, the word may be returned to the array or routed to another process until the entire array has been processed. The key to this structure is the single word processing and the hardware minimization. Even with a sophisticated program counter, the machine is capable only of single word processing and is therefore inherently slow. Due to its special purpose nature, however, the serial architecture normally possesses a factor of five advantage in speed over the general purpose digital computer. Two considerations are important in the design of the actual structure. First, the manner in which the data array is formed and the processing philosophy i.e. one word processed in total on a word sequential basis or the same process to be performed on the entire array prior to selecting the next process. (Studies have shown the maximum speed and simplest control is achieved when one process is applied against the entire array prior to

FIGURE 5.1-3

SERIAL ARCHITECTURE



## SERIAL ARCHITECTURE

### ADVANTAGES

1. LIMITED HARDWARE REQUIREMENT
2. HIGH SYSTEM LEVEL EFFICIENCY
3. CAPABLE OF COMPLEX ALGORITHMS
4. ECONOMICAL
5. ELECTRONIC MODIFICATION OF SIGNAL FLOW

### DISADVANTAGES

1. LOW OPERATIONAL FREQUENCY
2. INEFFICIENT AT THE PROCESSING LEVEL
3. LIMITED GROWTH POTENTIAL
4. TIME-SHARED BUS ORIENTATION

TABLE 5.1-3

selecting the next process resulting in a fixed state and mode machine). Second, the physical location of the processing functions with respect to their neighbors as a function of the system processing. The physical location determines the transitions within the recirculation loop and the overall timeliness. To transition within two nearest neighbors and bi-directionally the rate at which the block of information is processed becomes high while the effective word rate remains low.

This structure normally employs a fixed program and has application for low frequency multiplexed sensors and single input medium frequency sensors.

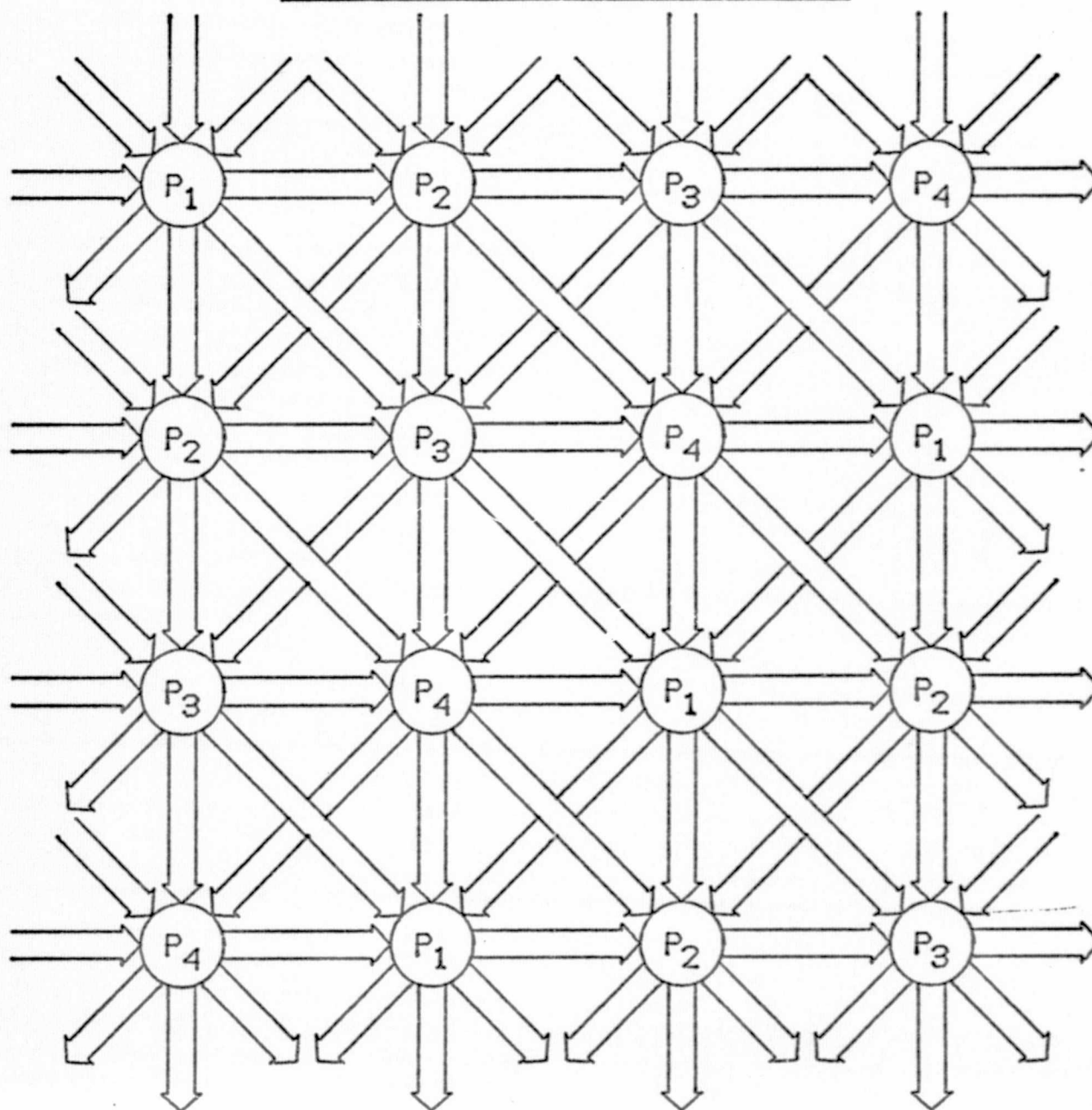
#### ARRAY

The array processor is the most sophisticated of the current information processors. The functional architecture is shown in Figure 5.1-4 and the general characteristics indicated in Table 5.1-4. The key feature of the array processor is that it is capable of simultaneously processing large arrays of data which require complex functions.

In operation the array becomes a set of pipelines programmable with respect to functions and sequence as a function of time may be viewed in the following manner. An array is formed which is  $N^2$  for  $N$  functions based on the spectrum of processing functions required for various system configurations. Each function repeats itself in the array  $N$  times along a diagonal. Each process is assigned at two dimensional indices or address and is capable of communicating with its eight nearest neighbors. Normally, any one process will be capable of receiving data from four neighbors and transmitting to four resulting in high processing power. The number of input and output points equals  $12(N-2) + 20$ ; thus, for  $N=4$ , there would be 22 input and 22 output points.

FIGURE 5.1-4

ARRAY ARCHITECTURE





## ARRAY ARCHITECTURE

### ADVANTAGES

1. CAPABLE OF COMPLEX ALGORITHMS
2. HIGH OPERATIONAL FREQUENCY ON LARGE ARRAYS OF DATA
3. SIMULTANEOUS WORD PROCESSING OF LARGE BLOCKS OF DATA
4. ELECTRONIC SIGNAL FLOW MODIFICATION
5. ELIMINATION OF FEEDBACK LOOPS
6. CONTROL AND PROGRAMMING SIMPLICITY

### DISADVANTAGES

1. MUST BE USED WITH LARGE ARRAYS OF DATA
2. COMPLEX FABRICATION
3. LOW GATE EFFICIENCY

TABLE 5.1-4

ORIGINAL PAGE IS  
OF POOR QUALITY

At the initial cycle, the data is routed to any of 2N functions where it is processed during the cycle. At the completion of the cycle new indices in pairs are ordered and the data output from each process is re-directed and entered into the next set of processes. The sequencing is continued processing multiple blocks of multiple inputs simultaneously, resulting in extreme speed. The array architecture is a mode oriented system that is in reality a hybrid of the serial and pipeline structures. Due to its block processing, this architecture is optimum for processing large volumes of data at high speed but extremely inefficient with respect to utilization of processing functions for short durations and low volumes of data.

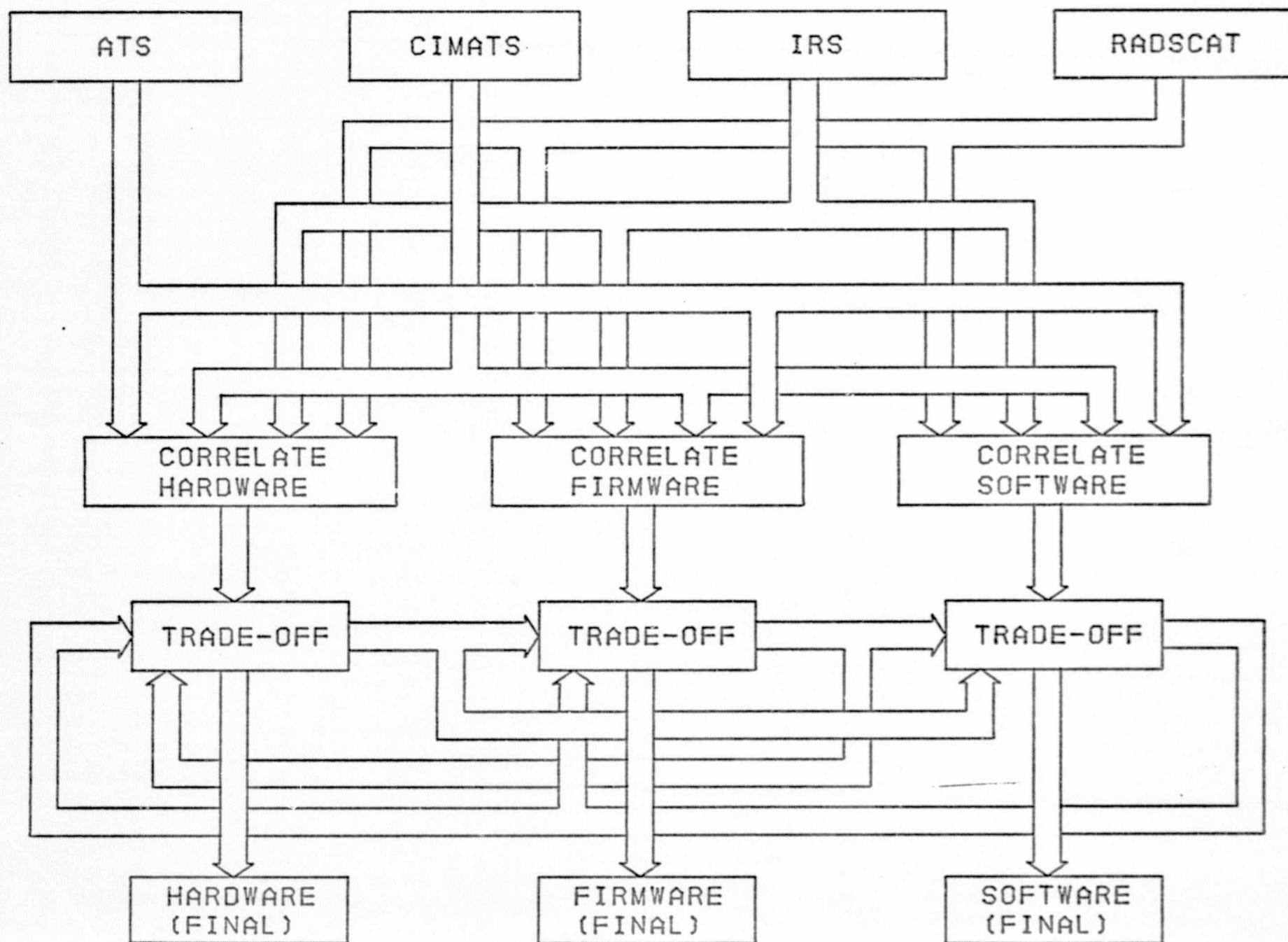
#### PROCESSING ARCHITECTURE TRADE-OFFS

The architectures discussed above are compared and a candidate architecture is selected for the subsystem level in this paragraph. The required OEDSF functions have been determined from the Functional Flow Diagram (Section 4.2). Based on decomposition and similarity of processing, the composite processing functions required reduce to those listed in Table 4.2-1. Figure 5.1-5 depicts the process followed to derive the composite system. Based on these requirements, the OEDSF must be capable of performing the following major functional categories:

- o Data Handling functions
- o Signal Calibration
- o Computational processing
- o Domain Transformations

The bulk of the majority of the functions is computational and domain transformations as expected for sensor processing.

ARCHITECTURE COMPOSITION TRADE-OFF APPROACH



For the majority of small computers, the addition at the register level requires approximately 2.0 microseconds and an additional 2.0 microseconds of overhead which is too slow to accommodate state-of-art sensors on a real time basis. However, with the current trends in computer architecture and semiconductor technology a factor of ten reduction will be achieved by the 1980 time frame so that the medium and low frequency sensors will be accommodated by the small computer. The pipeline and array processors are specifically oriented to high speed processing of complex algorithms. The serial processor, due to its special purpose nature, is above the general purpose machine but below the pipeline and array and is capable of processing a majority of the sensors.

Due to the number of permutations of sensor combination, the OEDSF must be capable of altering its process flow path. All the architectures except the pipeline are capable of accomplishing this electronically, however, only the array processor can achieve the alteration without loss of system speed. The pipeline, due to its sequential nature, requires a physical modification to re-configure the process flow.

In the sequence alteration, the pipeline is at an extreme disadvantage since some physical modifications once implemented are irreversible and are, further, extremely difficult to integrate and evaluate.

A significant aspect that must be considered is the implementation of the processes themselves. The small computer is software oriented so that implementation must be organized around the sub-routine level to gain the flexibility required in this type of system. Consequently, the modification capability is facilitated with the small computer and the evaluation can be made on the actual system.

The remaining three architectures require physical alteration. If the functions are modular and possess a standardized I/O, the modification is facilitated, however, the modification is still physical in nature which may or may not be a disadvantage depending on the actual design.

A comparison of the utilization of the electronic gating as a function of processing time is in order. This factor may be translated directly into electronic efficiency on an energy basis. The small computer with its many registers and general purpose approach is extremely inefficient in the utilization of its electronic gating. The very structure of the small computer forces many registers and devices to be dormant. Since the removal of these components destroys the computer, the efficiency at the register level is fixed and unalterable. Like the small computer, the serial architecture is low in electronic efficiency. Although some advantage is gained over the general purpose computer, the serial structure can only execute one operation at a time and therefore contains many dormant states. Since it is capable of operating at medium frequencies the information rate versus gate utilization becomes acceptable over an extended period of time. In addition, the serial architecture being special purpose in nature is capable of being modified.

The array architecture is a modular structure in nature and is characterized by a moderate to high gate utilization. The variable utilization occurs due to the inherent nature of the processor. First, the modularity allows for only the actual processes to be physically utilized. Second, the processor is designed to handle large blocks of information simultaneously but is dependent on the user to maximize the array processing functions. Dependent on these conditions, the processes may remain dormant like the small computer and serial architectures or may be exploited as in the pipeline. It must be reiterated that the efficiency

is centered on large arrays of data and that the structure is not oriented to processing small volumes of data.

The pipeline architecture optimizes gate utilization approaching 100 per cent. Since each word must pass through each stage, and during steady state the entire pipeline is configured with a word, the utilization is optimum.

An additional area of comparison is the user orientation. In order to achieve the necessary degree of flexibility, these architectures will be programmable and, therefore, require some programming language. In the small computer, the user is constrained to the language of the specific machine at both the assembly and procedural levels. If the machine family is capable of alteration (which it must be to maintain growth potential), the small augmented computer is not user oriented. Since the remaining structures are special purpose, a language will have to be developed. Although this may appear to be a monumental task, the architectures require only control languages rather than languages based on instruction sets.

Of the three structures, the array processor is the most advantageous. The language for the array processor is not only simple but process invariant due to the configuration by calling indices. A procedural language can be quickly developed to allow the user to program the array easily by calling source and destination indices. Regardless of the configuration, i.e. the processing functions located at the specific points on the matrix, the program language and technique will not change. Thus, the serial, pipeline, and array architectures are all favorably oriented toward the user with the latter being the most advantageous.

Finally, the interface capability must be examined. For the serial and pipeline structures only one point of entry and exit exists so that I/O is rather simple if a port standardization technique is employed. By port standardization is meant, the manner in which data is transferred between two machines and more specifically the protocol and timing. Once the port is standardized and documented, the interface is rather straightforward. For the small computer and the array multiple ports are available so that standardization is imperative and signal I/O must be carefully considered. Multiple port machines are extremely advantageous for OEDSF type data handling requirements, consequently, the array and small augmented computer architectures are applicable to the OEDSF while the serial and pipeline have drawbacks.

These factors are summarized on Table 5.1-5. The qualitative attributes were converted to numerical values using various scales (such as: Poor = 0, Fair - 1, Good = 2, Excellent = 3), the evaluation criteria were uniformly weighed. The array was a consistent winner and is the selected processing architecture.

## COMPARISON OF PROCESSING ARCHITECTURES

<u>EVALUATION CRITERIA</u>	<u>SMALL COMPUTER</u>	<u>SERIAL</u>	<u>PIPELINE</u>	<u>ARRAY</u>
ABILITY TO IMPLEMENT COMPLEX ALGORITHMS	EXCELLENT	GOOD	GOOD	EXCELLENT
OPERATIONAL SPEED AND COMPLEX PROCESSING	POOR	FAIR	EXCELLENT	EXCELLENT
SIGNAL FLOW MODIFICATION CAPABILITY	EXCELLENT	GOOD	POOR	GOOD
MODIFICATION OF PROCESSING FUNCTIONS	EXCELLENT	FAIR	FAIR	FAIR
GATE UTILIZATION EFFICIENCY	POOR	POOR	EXCELLENT	FAIR
INFORMATION PROCESSING EFFICIENCY ON LARGE ARRAYS	GOOD	GOOD	GOOD	GOOD
INFORMATION PROCESSING EFFICIENCY ON SMALL ARRAYS	POOR	FAIR	POOR	POOR
USER PROGRAMMING AND CONTROL ORIENTATION	GOOD	FAIR	FAIR	EXCELLENT
PHYSICAL ADAPTABILITY TO SPACECRAFT	EXCELLENT	EXCELLENT	EXCELLENT	EXCELLENT
INPUT/OUTPUT INTERFACE CAPABILITY	EXCELLENT	FAIR	GOOD	GOOD
SYSTEM DISTRIBUTION CAPABILITY	POOR	FAIR	EXCELLENT	EXCELLENT
DEVELOPMENT OF MACRO INSTRUCTION SET	GOOD	FAIR	EXCELLENT	POOR
MANUFACTURE AND TEST	GOOD	GOOD	EXCELLENT	POOR

TABLE 5.1-5



## 5.2 SYSTEM LEVEL ARCHITECTURES

This section discusses the applicable system level architectures and performs trade-offs to select a candidate architecture. As previously stated, the applicable architectures at the system level are

- o Centralized
- o Distributed
- o Structured

These are discussed at the conceptual level.

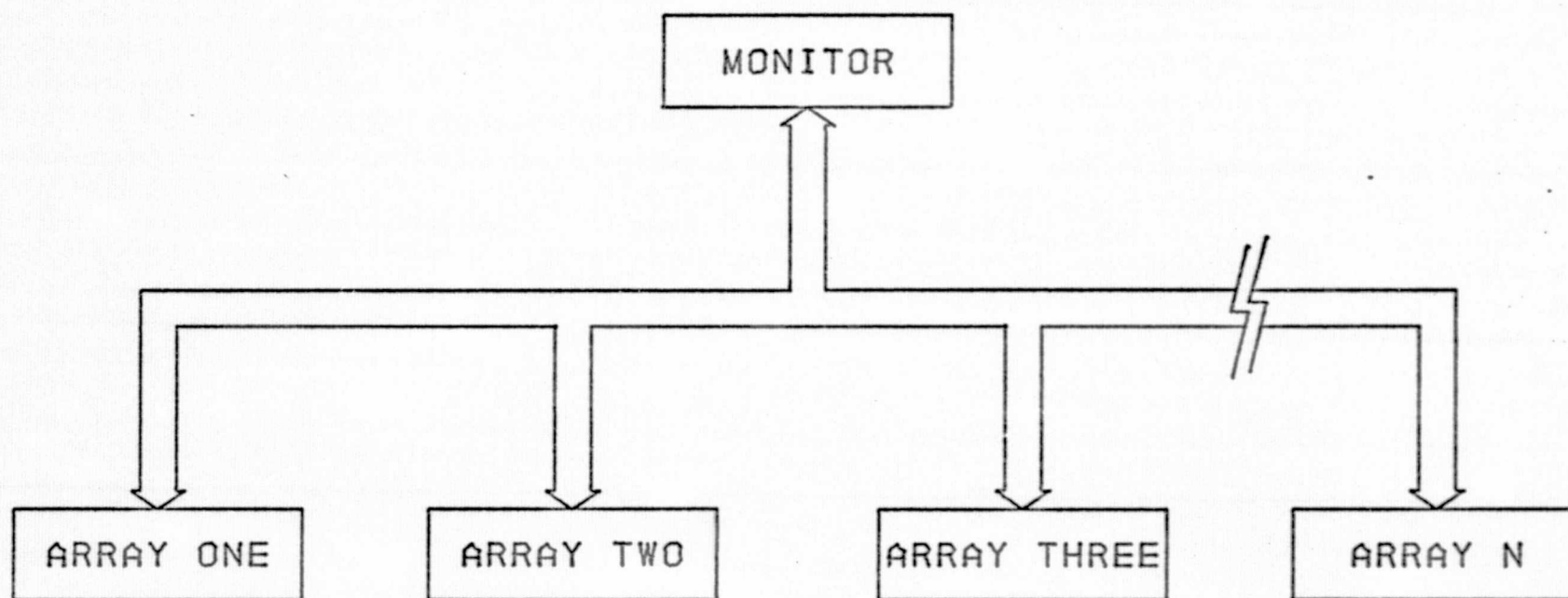
### CENTRALIZED ARCHITECTURE

The centralized architecture shown in Figure 5.2+1 is formulated on the philosophy of maintaining a single level of control over a system or set of subsystems.

In operation, a set of processing subsystems are employed to form a system with each subsystem under the direct control of the system monitor and is characterized by the absence of intermediate controllers. Due to the single level of monitoring, a rather complex control scheme may be required when the number of processing arrays used becomes large. Further, because of the single level of control, the monitor serves as a reference or synchronization point for the system and the control of a specific array may be determined in conjunction with the conditions or events present in other arrays or areas of concern that are external to the processing system itself such as the GNC system.

Consequently, the centralized architecture possesses an upper boundary on the number of processing arrays which may be used and on the control complexity required. A centralized architecture will therefore restrict an OEDSF type information processing sub-system.

FIGURE 5.2-1  
CENTRALIZED ARCHITECTURE



## DISTRIBUTED ARCHITECTURE

The distributed architecture shown in Figure 5.2-2 is formulated on a set of independent systems and sub-systems. Each subsystem possesses its own control and executes its scenario without regard for the state of the other subsystems. Consequently, this architecture is totally modular in nature enhancing growth potential and no limitation on the control complexity and number of processing arrays used. However, the independence between sub-systems prevents interaction which is anticipated for the Shuttle experiments data processing role. For example, one sensor may assess cloud cover so that its output will be the enabling or inhibiting event in another sensor such as the advanced technology scanner. Normally, this architecture is employed when total independence is permissible within the machine level.

## STRUCTURED ARCHITECTURE

The structured distributed architecture is shown in Figure 5.2-3. In reality it is a hybrid of the centralized and distributed. Basically, the control philosophy is to establish a pyramid which localizes the control responsibility. As the pyramid is ascended the control becomes more oriented towards the general system. As the pyramid is descended, the control becomes more oriented to the specific process or set of processes to be used. Consequently, independent arrays may be employed but subsystem interaction is available. This architecture is characterized by a high degree of modularity, growth potential, and simplicity.

## SYSTEM LEVEL ARCHITECTURE SELECTION

The structured architecture combines the advantages of the centralized and distributed architectures with little penalty. Since the OEDSF needs these features, the Structured ARchitecture is the selected system architecture.

FIGURE 5.2-2

DISTRIBUTED ARCHITECTURE

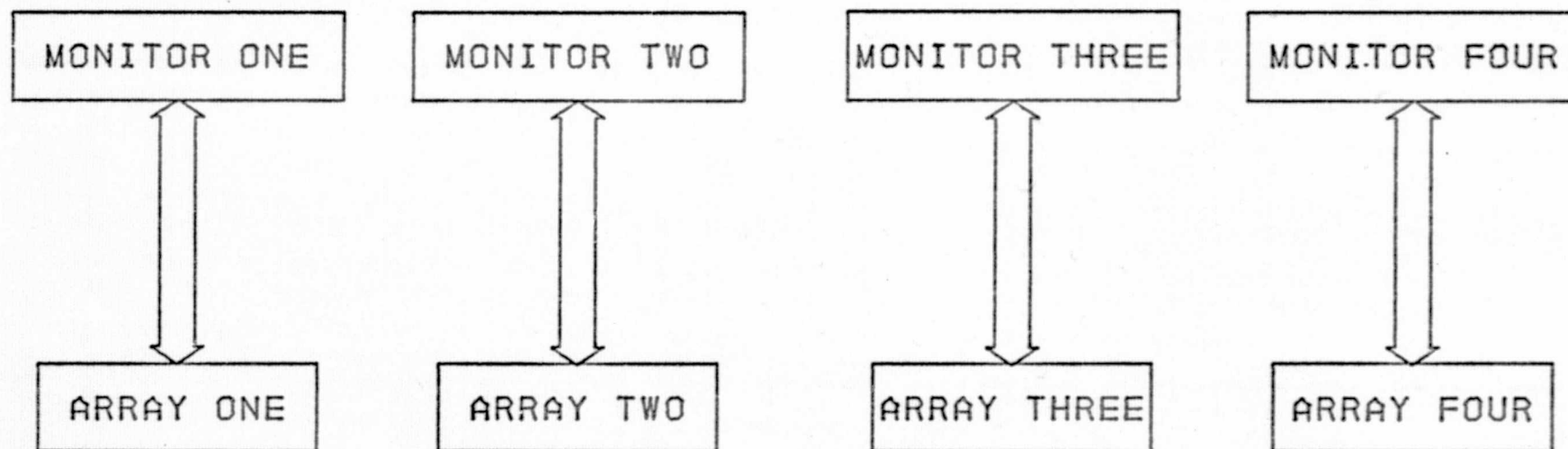
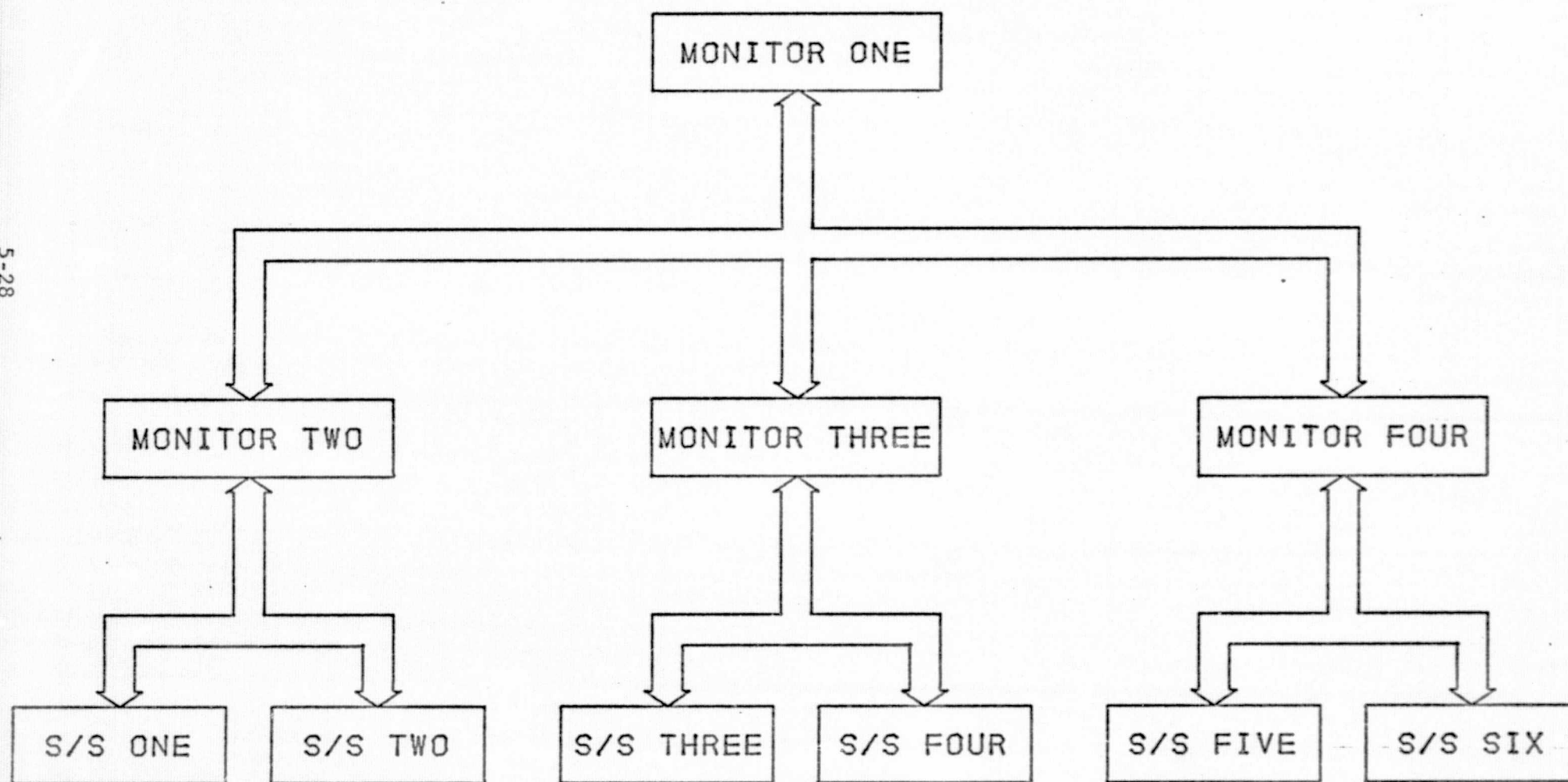


FIGURE 5.2-3

STRUCTURED ARCHITECTURE



APPENDIX A

## SINATS PROCESSES

### $\bar{a}(t)$ PROCESS

#### 1. REQUIRED PROCESS

$$\bar{a}(t) = \cos^{-1} [\sin \lambda_1(t) \sin \lambda_2(t) + \cos \lambda_1(t) \cos \lambda_2(t) \cos [\phi_2(t) - \phi_1(t)]]$$

#### 2. IMPLEMENTED PROCESS

$$\bar{a}(t) = \cos^{-1} [1 + \cot \lambda_1(t) \cot \lambda_2(t) \cos [\phi_2(t) - \phi_1(t)]]$$

#### 3. ALGORITHM

A.  $\bar{a}_1(t) = \cot \lambda_1(t)$

$\bar{a}_2(t) = \cot \lambda_2(t)$

$\Delta \phi(t) = \phi_2(t) - \phi_1(t)$

B.  $\bar{a}_3(t) = \bar{a}_1(t) \cdot \bar{a}_2(t)$

$\bar{a}_4(t) = \cos \Delta \phi(t)$

C.  $\bar{a}^*(t) = \bar{a}_3(t) \cdot \bar{a}_4(t)$

D.  $\bar{a}(t) = \cos^{-1} \bar{a}^*(t)$

#### 4. FUNCTIONS AND FREQUENCY REQUIRED

A.	COMPUTE COTANGENT	2
B.	COMPUTE COSINE	1
C.	COMPUTE SIGNED MULTIPLY	2
D.	COMPUTE ALGEBRAIC ADDITION	2
E.	COMPUTE INVERSE COSINE	1

ORIGINAL PAGE IS  
OF POOR QUALITY

## K(t) PROCESS

### 1. REQUIRED PROCESS

$$K(t) = \sin^{-1} \left[ \frac{\cos \lambda_2(t)}{\sin \alpha(t)} \sin [\phi_2(t) - \phi_1(t)] \right]$$

$$\sin K(t)$$

### 2. IMPLEMENTED PROCESS

$$K(t) = \sin^{-1} [\cos \lambda_2(t) \csc \alpha(t) \sin [\phi_2(t) - \phi_1(t)]]$$

$$\sin K(t)$$

### 3. ALGORITHM

A.  $b_1(t) = \cos \lambda_2(t)$

$b_2(t) = \csc \alpha(t)$

$\Delta \phi(t) = \phi_2(t) - \phi_1(t)$

B.  $b_3(t) = b_1(t) \cdot b_2(t)$

$b_4(t) = \sin \Delta \phi(t)$

C.  $b^*(t) = b_3(t) \cdot b_4(t)$

$b^*(t) = \sin K(t)$

D.  $K(t) = \sin^{-1} b^*(t)$

ORIGINAL PAGE IS  
OF POOR QUALITY

### 4. FUNCTIONS AND FREQUENCY REQUIRED

A. COMPUTE COSINE

B. COMPUTE COSECANT



c. COMPUTE SINE

1

d. COMPUTE ALGEBRAIC ADDITION

1

e. COMPUTE INVERSE SINE

1

f. COMPUTE SIGNED MULTIPLY

2

## $\lambda_x(t)$ PROCESS

### 1. REQUIRED PROCESS

$$\lambda_x(t) = \sin^{-1} [\cos \theta(t) \sin \lambda_1(t) + \sin \theta(t) \cos \lambda_1(t) \cos \kappa(t)]$$

### 2. IMPLEMENTED PROCESS

$$\lambda_x(t) = \sin^{-1} [1 + \tan \theta(t) \cot \lambda_1(t) \cos \kappa(t)]$$

### 3. ALGORITHM

A.  $c_1(t) = \tan \theta(t)$

$\bar{q}_1(t) = \cot \lambda_1(t)$   $\square$  REDUNDANT FUNCTION

B.  $c_2(t) = \cos \kappa(t)$

$c_3(t) = c_1(t) \cdot \bar{q}_1(t)$

C.  $c_4(t) = c_2(t) \cdot c_3(t)$

$c_5(t) = 1$

D.  $c^*(t) = c_4(t) + c_5(t)$

E.  $\lambda_x(t) = \sin^{-1} c^*(t)$

ORIGINAL PAGE IS  
OF POOR QUALITY

### 4. FUNCTIONS AND FREQUENCY REQUIRED

A. COMPUTE TANGENT

1

B. COMPUTE COTHINGENT

$\emptyset$

C. COMPUTE COSINE

1

D. COMPUTE SIGNED MULTIPLY

2

E. COMPUTE ALGEBRAIC ADDITION

1

F. COMPUTE INVERSE SINE

ORIGINAL PAGE IS  
OF POOR QUALITY

## $\phi_x(t)$ PROCESS

### 1. REQUIRED PROCESS

$$\phi_x(t) = \sin^{-1} \left[ \frac{\sin \kappa(t) \sin \theta(t)}{\cos \lambda_x(t)} + \phi_1(t) \right]$$

### 2. IMPLEMENTED PROCESS

$$\phi_x(t) = \sin^{-1} [\sin \kappa(t) \sin \theta(t) \sec \lambda_x(t) + \phi_1(t)]$$

### 3. ALGORITHM

A.  $d_1(t) = \sin \theta(t)$

$d_2(t) = \sec \lambda_x(t)$

$b^*(t) = \sin \kappa(t)$   $\square$  REDUNDANT FUNCTION

B.  $d_3(t) = d_1(t) \cdot d_2(t)$

C.  $d_4(t) = d_3(t) \cdot b^*(t)$

$d_5(t) = \phi_1(t)$

D.  $d^*(t) = d_4(t) + d_5(t)$

E.  $\phi_x(t) = \sin^{-1} d^*(t)$

### 4. FUNCTIONS AND FREQUENCY REQUIRED

A. COMPUTE SINE

B. COMPUTE SECANT

C. COMPUTE SIGNED MULTIPLY

D. COMPUTE ALGEBRAIC ADDITION

E. COMPUTE INVERSE SINE

## M(t) PROCESS

### 1. REQUIRED PROCESS

$$M(t) = 1 + \sec \Theta(t)$$

### 2. IMPLEMENTED PROCESS

$$\bar{M}(t) = \sec(\Theta(t)/2)$$

### 3. ALGORITHM

A.  $e(t) = \Theta(t)/2$       SHIFT RIGHT

B.  $e^*(t) = \sec e(t)$

### 4. FUNCTIONS AND FREQUENCY REQUIRED

A. SHIFT RIGHT

1

B. COMPUTE SECANT

1

ORIGINAL PAGE IS  
OF POOR QUALITY

## Q(t) PROCESS

### 1. REQUIRED PROCESS

$$Q(t) = \sum I(t) \cdot H(t)$$

### 2. IMPLEMENTATION PROCESS

$$Q(t) = \sum I(t) \cdot H(t)$$

### 3. ALGORITHM

A.  $f_1(t_n) = I(t_n) \cdot H(t_n)$

$$f_2(t_n) = \sum_{m=0}^{n-1} f_1(t_m)$$

B.  $f^*(t_n) = f_1(t_n) + f_2(t_n)$

### 4. FUNCTIONS AND FREQUENCY REQUIRED

A. COMPUTE SIGNED MULTIPLY

B. COMPUTE ALGEBRAIC ADDITION

## h(t) PROCESS

### 1. REQUIRED PROCESS

$$h(t) = [R + A(t)] \sin \phi(t) - R$$

### 2. IMPLEMENTED PROCESS

$$h(t) = [R + A(t)] \sin \phi(t) - R$$

### 3. ALGORITHM

A.  $g_1(t) = \sin \phi(t)$

$$g_2(t) = R + A(t)$$

B.  $g_3(t) = g_1(t) \cdot g_2(t)$

$$g_4(t) = \bar{R}$$

C.  $g^*(t) = g_3(t) + g_4(t)$

### 4. FUNCTIONS AND FREQUENCY REQUIRED

A. COMPUTE SINE

1

B. COMPUTE SIGNED MULTIPLY

1

C. COMPUTE ALGEBRAIC ADDITION

2

ORIGINAL PAGE IS  
OF POOR QUALITY

## $\Delta Q(t)$ PROCESS

### 1. REQUIRED PROCESS

$$\Delta Q(t) \leftarrow \max |Q_{NI}(t_n), Q_{NT}(t_n)|$$

### 2. IMPLEMENTED PROCESS

$$\Delta Q(t) = \Delta Q(t_{n-1}) [\Delta Q(t_{n-1}) > \Delta Q(t_n)] \oplus \Delta Q(t_n) [\Delta Q(t_n) > \Delta Q(t_{n-1})]$$

### 3. ALGORITHM

A.  $\dot{L}_1(t_n) = Q_{NI}(t_n) - Q_{NT}(t_n)$

B.  $\dot{L}_2(t_n) = \dot{L}_1(t_n) \oplus \dot{L}_1(t_{n-1})$        $\dot{L}_2(t_n) = 1$  FOR  $\dot{L}_1(t_n) > \dot{L}_1(t_{n-1})$

$\dot{L}_3(t_n) = \dot{L}_1(t_{n-1}) \oplus \dot{L}_1(t_n)$        $\dot{L}_3(t_n) = 1$  FOR  $\dot{L}_1(t_{n-1}) > \dot{L}_1(t_n)$

C.  $\dot{L}_4(t) = \dot{L}_1(t_n) \cdot \dot{L}_2(t_n)$

$\dot{L}_5(t) = \dot{L}_1(t_{n-1}) \cdot \dot{L}_3(t_n)$

d.  $\dot{L}^*(t) = \dot{L}_4(t) + \dot{L}_5(t)$

### 4. FUNCTIONS AND FREQUENCY REQUIRED

A. COMPUTE ALGEBRAIC ADDITION	2
B. COMPUTE MAGNITUDE COMPARISON	2
C. COMPUTE SIGNED MULTIPLY	2



## BASIC PROCESSES

### 1. REQUIRED PROCESSES

$$C(z) = (0.5) Q(z) \cdot A(z)$$

$$R(z) \Rightarrow R(z) + 1$$

### 2. IMPLEMENTED PROCESSES

$$C(z) = (0.5) \cdot Q(z) \cdot A(z)$$

$$R(z) \Rightarrow R(z) + 1$$

### 3. ALGORITHM

A. COMPUTE SIGNED MULTIPLY 1

B. COUNTER INCREMENT 1

### 4. FUNCTIONS AND FREQUENCY REQUIRED

NOT APPLICABLE

ORIGINAL PAGE IS  
OF POOR QUALITY

## $C_m(t)$ PROCESS

### 1. REQUIRED PROCESS

$$C_m(t) = \frac{\sum X_m^2(t) - [\sum X_m(t)] X_m(t)}{M \sum X_m^2(t) - [\sum X_m(t)]^2}$$

### 2. IMPLEMENTED PROCESS

$$C_m(t) = \frac{\sum X_m^2(t) - [\sum X_m(t)] X_m(t)}{M \sum X_m^2(t) - [\sum X_m(t)]^2}$$

### 3. ALGORITHM

A.  $a_1(t) = \sum X_m^2(t)$

$a_2(t) = \sum X_m(t)$

B.  $a_3(t) = M \cdot a_1(t)$

$a_4(t) = a_2(t)^2$

$a_5(t) = a_2(t) \cdot X_m(t)$

C.  $a_6(t) = a_1(t) - a_5(t)$

$a_7(t) = a_3(t) - a_4(t)$

D.  $a_8(t) = a_7^{-1}(t)$

E.  $a^*(t) = a_6(t) \cdot a_8(t)$

### 4. FUNCTION AND FREQUENCY REQUIRED

A. COMPUTE SIGNED MULTIPLY 3

B. COMPUTE ALGEBRAIC ADDITION 2

C. COMPUTE ALGEBRAIC ACCUMULATION 4

## KALMAN FILTER PROCESS

### 1. REQUIRED PROCESS

$$\hat{a}_s(t) = \hat{a}_s(t_{n-1}) + w(t_n) [\hat{a}(t_n) - \hat{a}_s(t_{n-1})]$$

### 2. IMPLEMENTED PROCESS

$$\hat{a}_t(t) = \hat{a}_s(t_{n-1}) + w(t_n) [\hat{a}(t_n) - \hat{a}_s(t_{n-1})]$$

### 3. ALGORITHM

A.  $c_1(t) = \hat{a}(t_n) - \hat{a}_s(t_{n-1})$

$$c_2(t) = w(t_n)$$

B.  $c_3(t) = c_1(t) \cdot c_2(t)$

$$c_4(t) = \hat{a}_s(t_{n-1})$$

C.  $c^*(t) = c_4(t) + c_3(t)$

### 4. FUNCTION AND FREQUENCY REQUIRED

A. COMPUTE ALGEBRAIC ADDITION

2

B. COMPUTE SIGNED MULTIPLY

1

## GAIN AND OFFSET PROCESS

### 1. REQUIRED PROCESS

$$a(z) = \sum_{m=0}^{N-1} c_m(z) \cdot v_m(z)$$

$$b(z) = \sum_{m=0}^{N-1} c_m(z) \cdot v_m(z)$$

### 2. IMPLEMENTED PROCESS

$$g(z) = \sum_{m=0}^{N-1} c_m(z) \cdot v_m(z)$$

### 3. ALGORITHM

$$A. \quad b_1(z_n) = c_m(z_n) \cdot v_m(z_n)$$

$$b_2(z_n) = c_m(z_{n-1}) \cdot v_m(z_{n-1})$$

$$B. \quad b^*(z_n) = b_1(z_n) + b_2(z_n)$$

### 4. FUNCTION AND FREQUENCY REQUIRED

A. COMPUTE SIGNED MULTIPLY 1

B. COMPUTE ALGEBRA ADDITION 1

ORIGINAL PAGE IS  
OF POOR QUALITY

## LINEAR PROCESS

### 1. REQUIRED PROCESS

$$V_c(t_n) = a(t_n) V_{IN}(t_n) + b(t_n)$$

### 2. IMPLEMENTED PROCESS

$$V_c(t_n) = a V_{IN}(t_n) + b \quad \text{FOR } f_n > f_1$$

### 3. ALGORITHM

A.  $d_1(t_n) = a$

$$d_2(t_n) = a V_{IN}(t_{n-1}) + b$$

B.  $d^*(t_n) = d_1(t_n) + d_2(t_n)$

### 4. FUNCTION AND FREQUENCY REQUIRED

A. COMPUTE ALGEBRAIC ADDITION

1

## HADAMARD TRANSFORM PROCESS

### 1. REQUIRED FUNCTION

$$H(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) \cdot (-1)^{P(x, y, u, v)}$$

$$P(x, y, u, v) = \sum_{i=0}^{N-1} (u_i \cdot x_i + v_i \cdot y_i)$$

### 2. IMPLEMENTED FUNCTION

$$H(u, y) = \sum_{x=0}^{N-1} f(x, y) \cdot (-1)^{P(x, y)}$$

$$H(u, v) = \sum_{y=0}^{N-1} H(u, y) \cdot (-1)^{P(y, v)}$$

$$P(x, y) = \sum_{i=0}^{N-1} u_i x_i$$

$$P(y, v) = \sum_{i=0}^{N-1} v_i y_i$$

### 3. ALGORITHM

$$A. e_1(t) = \sum_{i=0}^{N-1} u_i x_i$$

$$e_2(t) = \sum_{i=0}^{N-1} v_i y_i$$

$$B. e_3(t) = (-1)^{e_1(t)}$$

$$e_4(t) = (-1)^{e_2(t)}$$

$$C. e_5(t) = f(x, y) \cdot e_3(t)$$

$$D. e_6(t) = \sum e_5(t)$$

$$E. e_7(t) = e_6(t) \cdot e_4(t)$$

$$F. e^*(t) = \sum e_7(t)$$

ORIGINAL PAGE IS  
OF POOR QUALITY

### 4. FUNCTION AND FREQUENCY REQUIRED

A. COMPUTE SIGNED MULTIPLY

2

B. COMPUTE EXPONENT

2

C. COMPUTE ALGEBRIAC ACCUMULATION

4

ORIGINAL PAGE IS  
OF POOR QUALITY

## $K_d(t)$ PROCESS

### 1. REQUIRED PROCESS

$$K_d(t) = \frac{C_1 V_d}{\ln[C_2 V_d \cdot I(t)] + 1}$$

### 2. IMPLEMENTED PROCESS

$$K_d(t) = K_1 \cdot [\ln[K_2 \cdot I(t)] + 1]^{-1}$$

### 3. ALGORITHM

A.  $a_1(t) = K_2 \cdot I(t)$

$a_2(t) = 1$

B.  $a_3(t) = a_1(t) + a_2(t)$

C.  $a_4(t) = \ln a_3(t)$

D.  $a_5(t) = a_4^{-1}(t)$

$a_6(t) = K_1$

E.  $a^*(t) = a_5(t) \cdot a_6(t)$

### 4. FUNCTIONS AND FREQUENCY REQUIRED

A. COMPUTE SIGNED MULTIPLY

2

B. COMPUTE ALGEBRAIC ADDITION

1

C. COMPUTE NATURAL LOG

1

D. COMPUTE RECIPROCAL

1



## $B_J(t)$ PROCESS

### 1. REQUIRED PROCESS

$$B_J(t) = \frac{2hc^2 V_0(d)}{\exp[hV_0(d) \cdot [dK_c(t)]^{-1}] - 1}$$

### 2. IMPLEMENTED PROCESS

$$B_J(t) = \frac{2hc^2 V_0(d)}{\exp[hV_0(d) \cdot [dK_c(t)]^{-1}] - 1}$$

### 3. ALGORITHM

A.  $a_1(t) = d \cdot K_c(t)$

B.  $a_2(t) = a_1^{-1}(t)$

C.  $a_3(t) = hV_0(d) \cdot a_2(t)$

D.  $a_4(t) = \exp a_3(t)$

$$a_5(t) = 1$$

E.  $a_6(t) = a_4(t) + a_5(t)$

F.  $a_7(t) = a_6^{-1}(t)$

$$a_8(t) = 2hc^2 V_0(d)$$

G.  $a^*(t) = a_7(t) \cdot a_8(t)$

### 4. FUNCTION AND FREQUENCY REQUIRED

A. COMPUTE SIGNED MULTIPLY

3

B. COMPUTE ALGEBRAIC ADDITION

1

C. COMPUTE EXPONENTIAL . 1

D. COMPUTE RECIPROCAL 1

## MATRIX MULTIPLY PROCESS

### 1. REQUIRED PROCESS

$$\hat{P} = \begin{vmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \\ a_{13} & a_{23} & a_{33} \end{vmatrix} \times \begin{vmatrix} b_{11} & b_{21} & b_{31} \\ b_{12} & b_{22} & b_{32} \\ b_{13} & b_{23} & b_{33} \end{vmatrix}$$

### 2. IMPLEMENTED PROCESS

$$T(t) = \begin{vmatrix} a_{11}(t) & a_{21}(t) & a_{31}(t) \\ a_{12}(t) & a_{22}(t) & a_{32}(t) \\ a_{13}(t) & a_{23}(t) & a_{33}(t) \end{vmatrix} \times \begin{vmatrix} b_{11}(t) & b_{21}(t) & b_{31}(t) \\ b_{12}(t) & b_{22}(t) & b_{32}(t) \\ b_{13}(t) & b_{23}(t) & b_{33}(t) \end{vmatrix}$$

### 3. ALGORITHM

SINCE MATRIX MULTIPLICATION IS STANDARD AND PRESENTED IN ANY MATHEMATICAL HANDBOOK NO ALGORITHM WILL BE PRESENTED REFERENCE CRC TABLES ANY EDITION.

### 4. FUNCTIONS AND FREQUENCY REQUIRED

A. COMPUTE SIGNED MULTIPLY 27

B. COMPUTE ALGEBRAIC ADDITION 18

ORIGINAL PAGE IS  
OF POOR QUALITY

APPENDIX B

## HARDWARE/FIRMWARE/SOFTWARE

Hardware, firmware and software are terms used routinely by data processing and digital systems engineers. These may not, however, be necessarily familiar, in terms of exact meaning, to those not actively engaged in the design of digital hardware. In order to alleviate this potential difficulty and to standardize the meanings of these terms for subsequent discussions, an attempt will be made to define, illustrate and compare the advantages and disadvantages of hardware, firmware and software.

Since hardware, firmware and software are terms usually applied to some aspect of computer-type processor design, and since these terms evolved as a function of historical computer development, it would be profitable to examine the structure of a computer.

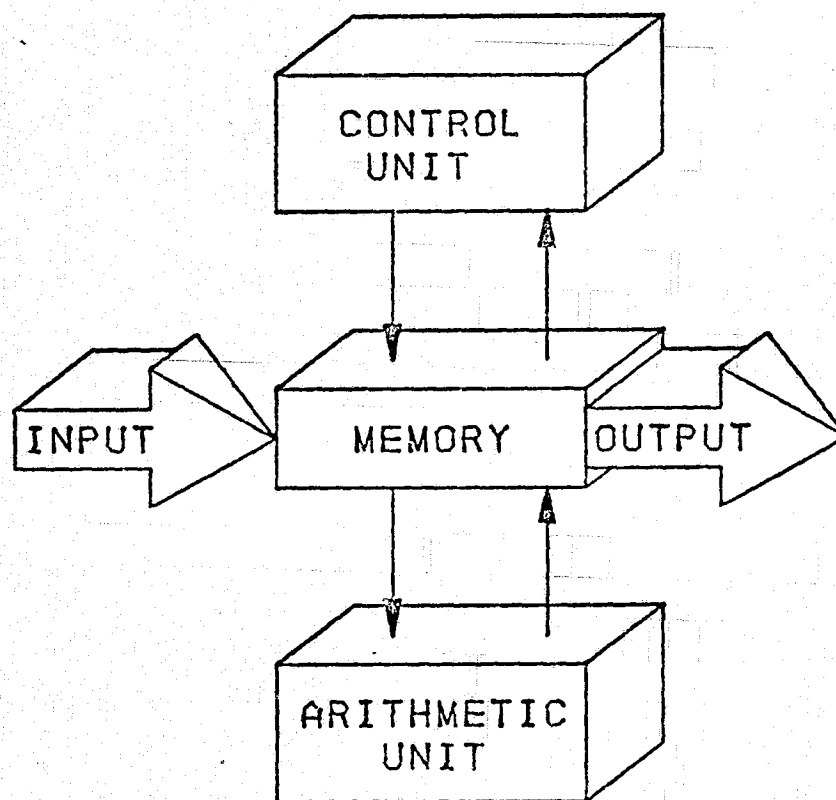
The structure and information paths of a typical fixed instruction stored program computer are represented in the simplified block diagram in Figure B-1 as defined in most textbooks, the five elements comprising a digital computer are: memory, arithmetic unit, input, output and control unit. Of particular relevance to the present discussion is the control unit.

The control unit may be referred to as the "brain" portion of any computer because it coordinates all units of the computer in timed logical sequence. The control unit of a small fixed instruction computer receives sequences of instructions, called macroinstructions, from memory. These sequences of macroinstructions called programs, reside in the memory and are referred to as software. The control unit is closely synchronized to the macroprogram memory speed and the execution time of each fixed instruction is usually a multiple of the memory speed.

SIMPLIFIED BLOCK DIAGRAM

FIXED INSTRUCTUIN STORED PROGRAM GENERAL-PURPOSE COMPUTER

FIGURE B-1



In the past, computer designers have specified control logic as complex sets of hard-wired connections for executing all machine instructions. As an alternative, the control unit can be configured as a memory. A sequence of subcommands or microinstructions is stored in this memory, to define each complete machine instruction.

The elements of a microprogrammed computer are shown in Figure B-2. The basic control sequences are stored in a separate control memory, usually a read-only memory which operates many times faster than the main memory section of the computer, instead of being specified in hard-wired logic as in the fixed instruction computer. The sequence of control instructions is called a microprogram. This is analogous to the way in which software is specified, as sequences of machine instructions stored in memory. Machines which utilize this organizational approach are stored logic or microprogrammed computers, and the memories used for microprograms are control stores. The program stored in the control store is called firmware.

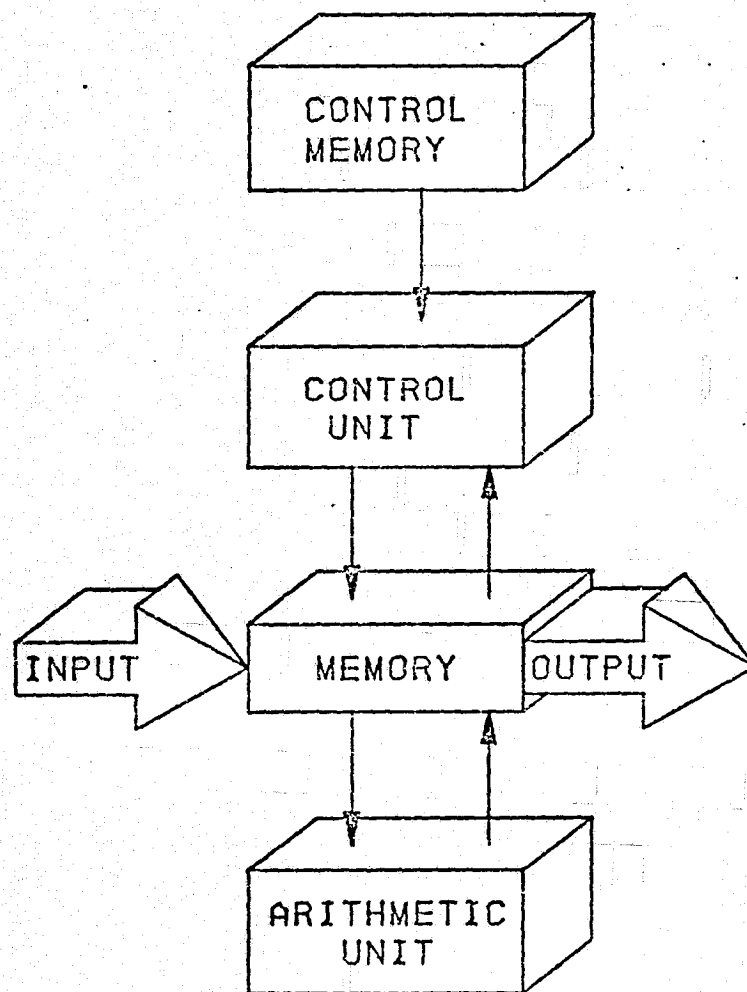
Microprogramming can be considered the process of specifying machine macroinstruction sets as sequences of microinstructions. The resultant microprograms or firmware determine the behavior of the basic computer hardware to make it perform particular functions. This contrasts with "hardware" which involves pre-wired equipment and is not altered after delivery, and software which implies using computer programs stored in main memory and comprised of macroinstructions to achieve desired performance.

Although the original intent of microprogramming was to simplify control unit design, the implications have been more far-reaching. For example, the structure of the control unit has become independent of the instruction set being implemented. By changing the contents of the control store, it is therefore possible to modify an instruction set late in the development of the hardware, or to select codes to suit the individual requirements of a program.

In the following discussion it is intended to compare firmware with hardware and software in such categories as speed, cost, reliability and flexibility. The hardware/software tradeoff will not be discussed separately as this will be implicit in the other comparisons.

SIMPLIFIED BLOCK DIAGRAM  
MICROPROGRAMMED COMPUTER

FIGURE B-2





## FIRMWARE VERSUS SOFTWARE

The software functions best suited to replacement by microprograms are those which are limited by CPU capabilities rather than by input/output transfer time, and those which produce intermediate results which are necessary for processing but not for final data. Highly repetitive functions, those which are awkward to do with an existing instruction set, and operations with large proportions of overhead time may also be improved by firmware. Complex computations which use basic machine instructions are less likely to be improved with microprogramming.

Firmware also makes it possible to add capabilities which are impractical in software, such as bootstrap loading to initialize operation of a bare machine. In addition, by making an entire function a single microprogram sequence, functions can be made indivisible with respect to interruption. This may be important on critical multiple-instruction operations, in which the integrity of a function may be destroyed by temporary suspension.

Implementing microprograms is not unlike producing software, except that simulators and other aids to microprogramming are not yet at the sophistication level of software techniques. Further the microprogrammer must have a good understanding of the hardware, including considerations of timing between instructions and support for parallel execution of operations by a single instruction.

### Speed

Microprogramming offers speed increases over software, which can enhance system performance. However replacing software functions by microprograms does not necessarily reduce execution time, and the actual performance ratio depends on the operations to be performed as well as the characteristics of the microinstruction set.

Microprogram performance is largely determined by the ratio of microinstruction to main memory speed. When memory access for data is not required, this ratio roughly indicates the number of microinstructions which can be executed during each instruction cycle. Speed is also enhanced by use of variable-length microinstruction operands. Hardware is often able to support different operand sizes, although a single length may be specified at the machine instruction level for convenience in defining a general-purpose instruction set. The microinstruction set may have more complete access to processor registers to take better advantage of this factor.

Other efficiency gains may accrue if a single microprogram can perform the function of many software instructions, since fetch and decode time are reduced. In addition, the associated decrease in software storage requirements can improve memory utilization. Firmware can also replace frequently-used subroutines. This eliminates the execution time needed to link to and from the subroutine in software, as well as the memory space of the subroutine. This can result in a significant performance improvement.

Further, special increases are possible if constants are incorporated in the microprograms, since the number of memory accesses for fetching and storing these values can be reduced. Similar gains are possible if intermediate results are stored in fast hardware registers or scratchpad memories accessible to the microinstructions.

Finally, microinstructions permit coding of more efficient algorithms, although the actual efficacy of microprogramming as an alternative to software depends both on hardware resources and the algorithms chosen for application. As an example, a square root operation was coded in software using the Newton-Raphson method, and was microprogrammed using an algorithm which computed results one bit at a time. The latter was eight times faster for 16-bit data. Similarly, microprogrammed implementation of a matrix inversion for two models of the IBM 360 gave a 3:1 improvement over equivalent software. On the other hand, a microprogrammed square root algorithm on a hypothetical machine with writable microstorage was only 20% faster than the equivalent program in IBM 360 assembly language. In another study, improvement for microprogrammed composite arithmetic operations was only by a factor of 1.5, because the instruction fetch time saved was a small portion of the time spent on arithmetic operations.

### Reliability

Microprograms tend to be well protected from modification or destruction. For example read-only memories cannot be modified by computer operations, and many writable control stores can only be loaded with special microprograms. These factors tend to enhance system reliability.

If it is physically impossible to alter the contents of a control memory, vital algorithms and data cannot be compromised. Software rules can be imposed to restrict certain operations, but microprograms can more positively prevent undesirable functions from being performed. This keeps users from accessing or modifying restricted areas of memory, and therefore protects system software and prevents independent users from interfering with each other.

Firmware can also be effective in maintaining hardware reliability. For example, microdiagnostics can be implemented which require relatively few hardware circuits and therefore can be executed with only a small portion of the machine working. In addition, microinstructions dealing with hardware can disable parts of a system to localize errors. In cases where hardware is not accessible through microinstructions, or the control store is too small for complete diagnostics programs, a combination of firmware and software may be feasible. The application of microdiagnostics during system execution can also result in fail-soft operation, by detecting hardware failures and substituting alternate means of achieving required functions.

### Cost

At present, software procurement can represent as much as 70% of system development cost. Economics can therefore be achieved using microprograms rather than software. This reduces the amount of software to be produced, but in performing an evaluation, the cost of firmware development must not be overlooked.

A more subtle use of microprogramming to reduce software cost is to change machine architecture to more closely match the way in which the software designer would like to use a machine. Software can then be simplified, making programmers more productive and reducing coding errors. For example, functions such as storage and process management can be removed from the responsibility of the programmer and placed in the

microprogram. Similarly architectural changes are possible which can facilitate checking of software, with corresponding reductions in debugging time and cost.

When it is necessary to replace a computer, software is generally redesigned to take advantage of the new machine architecture. However if existing software is extensive, it may be economical to use microprogramming to make the new hardware emulate the instruction set of the machine for which the software was written. The emulation in many cases yields lower execution times than were obtained on the original machine.

## FIRMWARE VERSUS HARDWARE

A hardware designer can view the control unit in a microprogrammed computer as a matrix, with a vertical line for each control gate and a set of horizontal lines having access to all gates. It is then possible to specify a machine instruction as a sequence of horizontal lines, each of which is connected to the combination of gates needed to create the desired data path. Each horizontal line corresponds to a micro-operation and each sequence can be considered a microprogram for a complete machine instruction.

### Speed

Execution time for hard-wired functions is usually shorter than that for the equivalent operations executed in microcode. This is because hardware functions can usually be performed at memory speed, while the time associated with microcode depends on factors such as the number of microinstructions which can be executed in one memory cycle. Further, the hard-wired approach allows some operations to occur concurrently through parallel commands, while microinstructions must be executed in sequence.

### Flexibility

Microprogramming permits the functions performed by general-purpose hardware to be modified or customized. This provides a degree of flexibility, which can be useful in meeting unanticipated requirements such as addition of new peripheral devices, or in upgrading performance capabilities. An example of the latter is adding microcode for floating-point arithmetic. Microprogramming flexibility also makes it practical to standardize procurement, since a general-purpose hardware design can be specialized for use in different applications or environments. Similarly, microprogramming allows the operating modes of a single computer to be changed, to serve more than one purpose or to combine functions which would normally be performed by several different specialized processors.

## Reliability

Certain functions can be implemented using simple hardware logic, and therefore offer high reliability. However as complexity increases, the control logic in a microprogrammed computer becomes relatively simpler and therefore inherently more reliable than that encountered in hard-wired machines.

In addition when one hardware design can be microprogrammed to serve different functions, a facility may contain fewer total machines or several identical machines. This reduces maintenance problems, since identical machines microprogrammed to perform different functions are interchangeable at the hardware level. Each machine is therefore a potential replacement for one which has failed, and the result is an improvement in the total system reliability. Microprogramming can also contribute to improve reliability of operational systems by dynamically performing error check and correction, and may aid in achieving fail-safe operations by dynamically changing algorithms to bypass failed hardware components.

## Cost

The regular structure of microprogrammed control units will reduce the cost of hardware, by exploiting LSI technology which depends on high volume use of the same logic components. Microprogramming can also reduce the cost of changes, since new capabilities can often be added with no additional hardware logic.

Savings may also be realized by replacing several hardware units with a single processor, microprogrammed to perform multiple functions. This avoids duplication of elements such as power supplies, and permits common functions to be implemented just once. Another problem is that control and working stores for microprograms are usually expensive. Units are therefore often small, and difficulties arise from the complexity introduced by limited program and data space. A drawback to microprogramming also arises in read-only control stores, since modifications are usually accomplished by patching in a way which minimizes the number of physical changes but obscures the logic. Writable control stores and trends toward improved languages and development support tools should correct many of these difficulties.

## APPLICATION EXAMPLES

Firmware may be integrated with hardware and software to yield a total solution to a specific problem or class of problems. Some specific examples of applications which have benefitted from the use of microprogramming will show the effect of distributing appropriate functions among all three of the resources available to systems designers.

### Input-Output Processing

Applications of microprogramming to communications processors and device controllers are becoming common. Input-output processing usually deals with the conventions for handling specific devices, and must be responsive to interrupts and critical timing constraints. Simple functions are performed such as polling, message routing, assembling characters, formatting and error handling. Efficient processing can be done at the channel level in real-time by either the central processor or a separate miniprocessor. Microprogramming is well suited to this type of processing and is more easily modified than special-purpose processors.

### Real-Time Processing

Real-time applications are typified by data collected from sensors, and used for functions such as closed-loop control, signal correlation, spectral analysis, pattern recognition, and filtering. Large volumes of data are frequently encountered and processing efficiency is therefore important to achieve satisfactory response time. Further, repetitive operations are often performed on arrays of data, so that many entries must be accessed to perform one processing cycle.

When microprogramming is used as a direct substitute for software, performance gains are only on the order of 3:1 because hardware must perform the actual arithmetic operations. Significantly greater improvements are possible when microprogramming is used to take advantage of special-purpose hardware, such as high-speed floating-point multiply units. For example in a microprogrammed array processor, complex operations such as convolving multiply were 250 times faster. The microprogramming permitted the addition of special-purpose hardware without perturbing the behavior of the rest of the system hardware and software.

In a signal array processing application, execution time in software required 545 milliseconds per cycle when implemented in software. A combination of microprogramming and special hardware reduced the time to 49 milliseconds.

#### Computer System Modifications

Microprogramming is advantageous for integrating changes in machine architecture into existing operating environments. For example hardware configurations may change as new types of peripheral devices become available, software may change to meet new functional requirements, and improvements may be necessary because a system speed and capacity are found to be too low. A major consideration in such cases is to preserve existing software, because of the investment it represents.

Microprogramming can be particularly useful in adding to an existing computer, since machine operation can be modified while much of the original software continues to operate. For example by replacing selected software with microprogrammed instructions, increased speed or capabilities can be obtained for given applications. In addition, it is possible to retain an instruction set, but modify the algorithms for accomplishing instructions. This technique can be used to increase efficiency or add new functions which are transparent to the user.

Architectural changes to a computer can also be made, which provide new instruction set primitives appropriate to some class of applications. For example addressing schemes or classes of operands might be modified to better utilize the primitives employed in the application and better map requirements onto the computing resources. This might include incorporating provision for directly executing some statements in a higher-level language.

New modes of operation can be implemented by adding microprograms which create different instructions and a different architecture. This makes it possible to leave the original machine and its software unchanged, yet extend computer utility through new user programming capabilities. On small microprogrammable machines, the modes of operation can be switched by physically changing read-only memories. Alternatively, writable control stores can be used for dynamically changing modes or permitting concurrent operation in different modes.



Operational computer-based systems are often modified to increase processing capabilities. Difficulties are imposed when the machine is obsolete, since it may be impractical or impossible to acquire new computers of the same type or to provide additional capability on existing devices. An alternative is to use microprogrammable hardware, which can provide means for smoothing transitions and optimizing the cost and performance of the new equipment. The new machine can be configured to emulate the instruction set of the previous computer and accept existing software. It may also be possible to extend the capabilities of the new machine so that existing software can gradually be phased over to a new architecture. Addition of processors is another approach to adding capacity or decreasing processing time, and a new processor may be made to emulate an existing system through firmware.

## SUMMARY

In summary, a performance matrix summarizing the general attributes of hardware, firmware and software is presented in Table 1. It is very important to remember, in light of the detailed preceding discussion, that these are simply the overall and generalized rankings within the categories listed. These rankings may require modification for specific applications as pointed out in the discussion.

Performance Matrix

Table 1

	<u>Speed</u>	<u>Flexibility</u>	<u>Reliability</u>	<u>Cost</u>
Hardware	Very Hi	Lo	Hi	Hi
Firmware	Hi	Very Hi	Hi	Med
Software	Lo	Hi	Med	Hi